UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE ENG. DE COMPUTAÇÃO E AUTOMAÇÃO

CONTROLADORES LÓGICOS PROGRAMÁVEIS

Professor: Anderson Luiz de Oliveira Cavalcanti
Autor: João Teixeira de Carvalho Neto
Natal-RN, Novembro 2011

SUMÁRIO

1	**INTRODUÇÃO**	**1**
1.1	AUTOMAÇÃO INDUSTRIAL	1
1.1.1	CONCEITOS IMPORTANTES NA AUTOMAÇÃO INDUSTRIAL	1
1.1.2	AUTOMAÇÃO INDUSTRIAL E SUAS TECNOLOGIAS ASSOCIADAS	4
1.2	CLP NO CONTEXTO DA AUTOMAÇÃO INDUSTRIAL	6
2	**FUNCIONAMENTO DE UM CLP E SEUS COMPONENTES**	**8**
2.1	PRINCÍPIO DE FUNCIONAMENTO DE UM CLP	8
2.2	COMPONENTES DE UM CLP	9
3	**SFC (GRAFCET)**	**13**
3.1	CONCEITOS IMPORTANTES	13
3.2	DESCRIÇÃO DOS SEUS ELEMENTOS	14
3.2.1	ETAPAS	14
3.2.2	ARCOS ORIENTADOS	14
3.2.3	AÇÕES	15
3.2.4	TRANSIÇÕES	17
3.2.5	RECEPTIVIDADE	18
3.3	REGRAS DE EVOLUÇÃO	19
3.4	ESTRUTURA SEQUENCIAL	21
3.4.1	SEQUENCIA ÚNICA	21
3.4.2	SELEÇÃO ENTRE SEQUÊNCIAS	21
3.4.3	CASOS PARTICULARES DE SELEÇÃO ENTRE SEQUENCIAS	22
3.4.4	SEQUÊNCIAS PARALELAS	22
3.5	EXEMPLOS UTILIZANDO GRAFCET	23
3.5.1	COMANDO DE UMA LÂMPADA POR UMA CHAVE ÚNICA	23
3.5.2	LIGAR UMA LÂMPADA DURANTE UM TEMPO	24
3.5.3	SISTEMA CARREGAMENTO DE UM CARRO	24
3.5.4	SISTEMA DE PARTIDA ESTRELA/TRIÂNGULO	25
3.5.5	SISTEMA DE UM PORTÃO AUTOMÁTICO	26
3.5.6	MÁQUINA DE FAZER BEBIDAS QUENTES	27
4	**LADDER (HI TECNOLOGIA)**	**29**
4.1	INTRODUÇÃO AO LADDER	29
4.2	INTRODUÇÃO AO AMBIENTE DE PROGRAMAÇÃO	31
4.3	ELEMENTOS DO EDITOR LADDER	32
4.3.1	INSTRUÇÕES E BLOCOS BÁSICOS	32
4.3.2	BLOCOS DE MOVIMENTO	38
4.3.3	BLOCOS DE MATEMÁTICA	41
4.3.4	BLOCOS DE COMPARAÇÃO	43
4.3.5	BLOCOS DE FLUXO	44
4.4	EXEMPLOS DE UTILIZAÇÃO DO LADDER	47
4.4.1	COMANDO DE UMA LÂMPADA POR UMA CHAVE ÚNICA	47
4.4.2	LIGAR UMA LÂMPADA DURANTE UM TEMPO	48
4.4.3	PARTIDA SEQUENCIAL DE MOTORES	48
4.4.4	SEGURANÇA NO COMANDO DE PRENSAS	49
4.4.5	SISTEMA DE PARTIDA DIRETA DE UM MOTOR	49
4.4.6	SISTEMA DE PARTIDA COM REVERSÃO DE UM MOTOR	50
5	**SUPERVISÓRIO INTOUCH**	**51**
5.1	INTRODUÇÃO AO SISTEMA SUPERVISÓRIO	51
5.2	GUIA SOBRE O INTOUCH	53
5.2.1	CRIANDO UMA APLICAÇÃO	53
5.2.2	INCLUINDO UMA APLICAÇÃO JÁ EXISTENTE	54
5.2.3	OUTRAS INFORMAÇÕES SOBRE A JANELA INTOUCH	55
5.3	GUIA SOBRE O WINDOWS MAKER	56
5.3.1	CRIANDO JANELAS	56
5.3.2	PROPRIEDADES DE UMA JANELA	56
5.3.3	DESCRIÇÃO DAS FERRAMENTAS	57

5.3.4 MENU DO WINDOWS MAKER 60
5.3.5 DESENVOLVENDO APLICAÇÕES 63
5.4 EXEMPLO DE UTILIZAÇÃO DO INTOUCH 69
5.4.1 CRIANDO UMA NOVA APLICAÇÃO 69
5.4.2 GRÁFICO DE BARRAS UTILIZANDO VARIÁVEL DO SISTEMA 70
5.4.3 BOMBA – BOTÃO LIGA/DESLIGA – SAÍDA COM ESTADO DA BOMBA 72
5.4.4 GRÁFICO DE BARRAS – VALOR ENTRADO PELO OPERADOR – BOTÃO 75
5.4.5 TEXTO – VISIBILIDADE E BLINK 76
5.4.6 DETECTOR DE GÁS – TAMANHO DO OBJETO 76
5.4.7 ESTEIRA 77
5.4.8 AJUSTE FINO PARA O GRÁFICO DE BARRAS 78
5.4.9 UTILIZANDO WIZARD – BOTÃO LIGA/DESLIGA 78
5.4.10 UTILIZANDO WIZARD – BOTÃO INCREMENTA/DECREMENTA 79
5.4.11 SCRIPT PARA ANIMAR A ESTEIRA 79
5.4.12 OBJETO FEITO COM OUTRO UTILITÁRIO E TRAZIDO COMO BITMAP 80
5.4.13 JANELA DE MENSAGEM PARA O OPERADOR 80
5.4.14 GRÁFICO DE TENDÊNCIA REAL 81
5.4.15 GRÁFICO DE TENDÊNCIA HISTÓRICA 81
5.4.16 JANELA DE ALARMES 82
5.4.17 BOTÕES PARA NAVEGAÇÃO ENTRE JANELAS 83
5.4.18 CRIAÇÃO DE GRUPOS DE ALARMES - PRIORIDADES 83
5.4.19 BOTÃO DE RECONHECIMENTO DE ALARMES 84
5.4.20 HISTÓRICO DE ALARMES – BOTÕES PAGUP E PAGDOWN 84

1- INTRODUÇÃO

1.1- AUTOMAÇÃO INDUSTRIAL

1.1.1- CONCEITOS IMPORTANTES NA AUTOMAÇÃO INDUSTRIAL

Para começar a entender os conceitos apresentados nesta apostila e saber como o que é um controlador lógico programável, é necessário entender o que é um controle, quais são seus elementos básicos e quais são os seus principais tipos.

Uma planta para ser controlada ela deve dispor dos seguintes componentes básicos: processo, sensor, controlador e atuador. Para o entendimento de o que é um controle é necessário o conhecimento destes componentes. O diagrama de um sistema de controle a seguir, apresenta estes elementos:

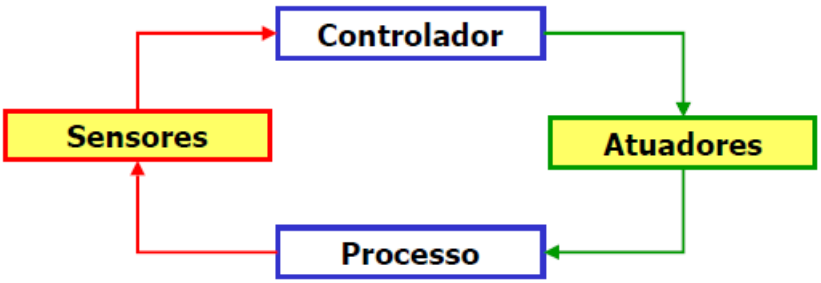

Figura 1.1 - Diagrama simplificado de um sistema de controle.

- **Processo:** operação que evolui progressivamente e que se constitui por uma série de ações controladas objetivando um resultado particular;
- **Sensor:** mede o desempenho do sistema de automação ou uma propriedade particular de algum de seus componentes. Exemplos: termômetros e velocímetros;
- **Controlador:** opera presença de distúrbios e tende a reduzir a diferença entre a saída de um sistema e a entrada de referência;
- **Atuador:** compara os valores medidos com valores preestabelecidos e toma a decisão de quando atuar no sistema. Como exemplos, podemos citar os termostatos e os programas de computadores, ou seja, são responsáveis pela realização de trabalho no processo ao qual está se aplicando a automação;

Sistemas de controle podem ser classificados basicamente em dois tipos:

- Sistema de controle em malha aberta: é aquele em que a saída ou resposta não possui nenhuma influência sobre a entrada.

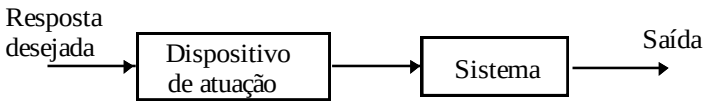

Figura 1.2 - Sistema de controle em malha aberta.

- Sistema de controle em malha fechada: é aquele em que a saída ou resposta influencia a entrada do sistema.

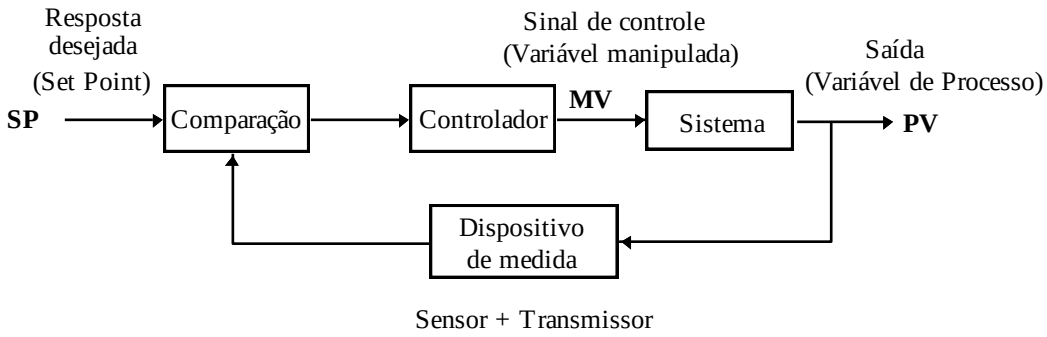

Figura 1.3 - Sistema de controle em malha fechada.

Desta forma, um sistema de malha fechada (principal foco desta apostila) é composto por um sensor que detecta a variável de processo (PV), um transmissor que converte o sinal do sensor em um sinal adequado e o transmite para um controlador que compara o valor da variável de processo (**PV**) com o valor do *Set Point* (**SP**) desejado e produz um sinal de controle apropriado e um elemento final de controle que muda a variável manipulada (**MV**). De acordo com a definição acima, percebe-se que a resposta de um sistema controlado é definida através de um programa do usuário e do comportamento das entradas e saídas. Assim é necessário citar os tipos de entradas e saídas que são utilizados na automação industrial.

- **Entradas discretas ou digitais**: fornecem apenas um pulso ao controlador, ou seja, A informação consiste em um estado binário da variável de controle (ligado ou desligado, por exemplo). Exemplos: botoeira, chave fim-de-curso.
- **Entradas analógicas**: a informação consiste em um dado que precisa ser digitalizado para ser processado, ou seja, Para trabalhar com este tipo de entrada os controladores tem conversores analógico-digitais. Exemplos: sensor de temperatura, sensor de nível.
- **Saídas discretas ou digitais**: exigem do controlador apenas um pulso que determinará o seu acionamento ou desacionamento. Exemplos: lâmpada de sinalização, relé.

- **Saídas analógicas**: como dito anteriormente, de forma similar o controlador necessita de um conversor digital para analógico, para trabalhar com este tipo de saída. Exemplos: válvula proporcional, conversor de frequência.

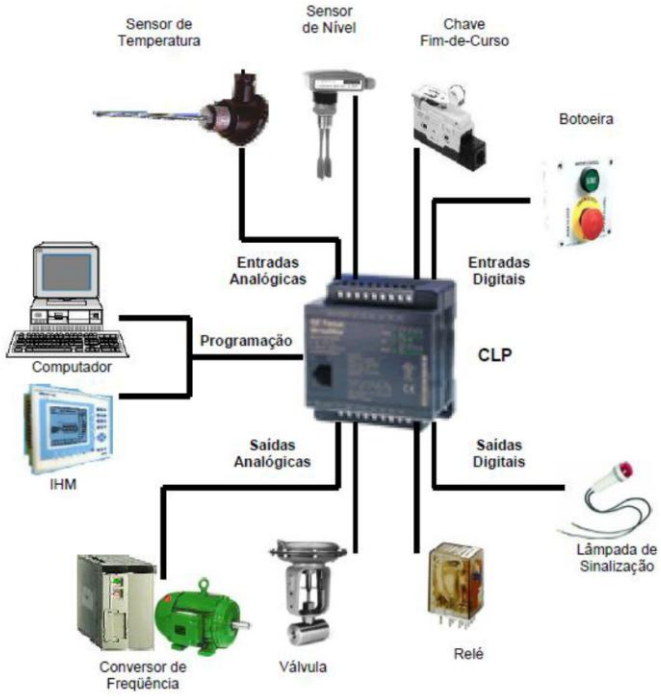

Figura 1.4 - Exemplos de entradas e saídas digitais e analógicas.

A seguinte figura ilustra um exemplo de um operador controlando a temperatura da água para o uso em um hotel (sistema de controle em malha fechada).

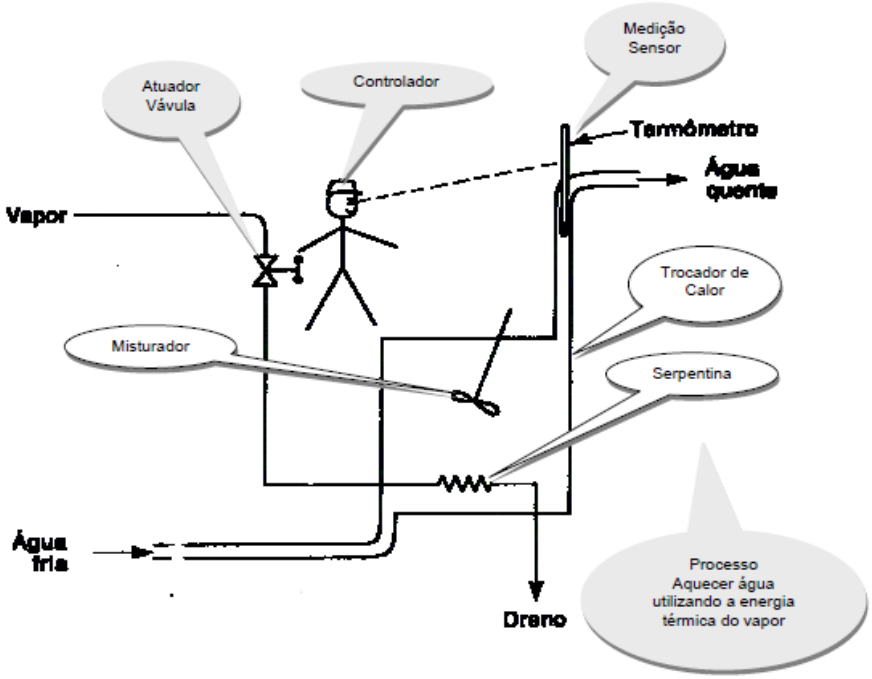

Figura 1.5 - Exemplo de um sistema de controle em malha fechada.

1.1.2- AUTOMAÇÃO INDUSTRIAL E SUAS TECNOLOGIAS ASSOCIADAS

As primeiras iniciativas do homem para mecanizar atividades manuais ocorreram na pré-história. Invenções como a roda, o moinho movido por vento ou força animal e as rodas d'água demonstram a criatividade do homem para poupar esforço. Porém, a automação só ganhou destaque na sociedade quando o sistema de produção agrário e artesanal transformou-se em industrial, a partir da segunda metade do século XVIII, inicialmente na Inglaterra. O controle manual implica em se ter um operador presente ao processo criador de uma variável física e que, de acordo com alguma regra de seu conhecimento, opera um aparelho qualquer (válvula, alavanca, chave, etc.), que por sua vez produz alterações naquela variável. A palavra automação está diretamente ligada ao controle automático, ou seja, está ligado às ações que dependem de forma mínima, da intervenção deste operador.

Dessa forma, com a finalidade de garantir o controle do sistema de produção, foram colocados sensores nas máquinas para monitorar e indicar as condições do processo. O controle só é garantido com o acionamento de atuadores a partir do processamento das informações coletadas pelos sensores.

Automatizar um sistema tornou-se muito mais viável à medida que a eletrônica avançou e passou a dispor de circuitos capazes de realizar funções lógicas e aritméticas com os sinais de entrada e gerar respectivos sinais de saída. Com este avanço, o controlador, os sensores e os atuadores passaram a funcionar em conjunto, transformando processo em um sistema automatizado, onde o próprio controlador toma decisões em função da situação dos sensores e aciona os atuadores.

Vale salientar dessa forma, que dois maiores objetivos ligados a não dependência do operador nos processos industriais é a redução dos custos da fabricação e a maximização da produtividade. Para isso existe uma pirâmide de níveis hierárquicos de um sistema industrial que ajuda a compreender como é possível atingir estes objetivos.

Na indústria, quando um grande pedido é emitido, já se conhece todo o controle do fluxo de materiais necessários para seu atendimento, ou seja, a disponibilidade de estoque, a genealogia das peculiaridades do produto, para que se possa prover a quantidade de matéria-prima utilizada para sua confecção, tempo de execução das máquinas em operação e datas de necessidade definidas para cada ponto ou etapa do processo; bem como o custo total estimado e prazo previsto de entrega do produto acabado. Para que este nível de integração seja possível, é necessário não só disponibilizar, como também compartilhar todos esses dados por meio das máquinas e/ou computadores que estiverem manipulando tais informações.

Aspirando também a uma redução de custos, o sistema passou a promover junto aos computadores, controladores e dispositivos digitais com capacidade de processamento autônomo de uma forma geral, uma intercomunicabilidade por meio físico adequado, de forma a obter grandes topologias dentro de regras gerais e procedimentos definidos para a transmissão de dados. Cria-se sistema de comunicação em rede e os computadores passam a trocar dados entre eles e a compartilhar recursos.

Quando informações de um sistema de controle são tratadas, a partir de dados coletados por meio de dispositivos sensores ou de qualquer dispositivo indicador de estado de um equipamento, por meio do processamento de dados feito por um sistema de supervisão, segundo um grande banco de dados, todos os níveis acima, como o de coordenação, planejamento e gerenciamento, são beneficiados de tal modo a melhor coordenar a programação e o controle de produção, planejando o processo de fabricação como um todo e estabelecendo uma nova gestão de gerenciamento dos negócios.

As grandes decisões dentro de um sistema organizacional de produção passam a ser tomadas dentro do mais alto grau do conceito da qualidade, baseado em dados concretos e atuais, que se originam nas mais diferentes unidades de controle do chão de fábrica. Informações precisas, envolvendo o processamento de um grande número de variáveis, são agentes sincronizadores do processo de fabricação, fazendo com que haja um planejamento global mais adequado, sustentado pelas limitações operacionais diante da capacidade de produção, regulando o nível de estoque, definindo e projetando produtos, estabelecendo estratégias e procedimentos de gerenciamento e diagnósticos.

Desse modo, verifica-se a importância de um sistema de comunicação integrada, estabelecida nos mais diferentes níveis de controle dentro da estrutura de recursos oferecidos por uma rede de processamento digital. Então, ao estabelecer a integração dos dados digitalmente por meio de uma rede de computadores entre os mais diferentes níveis hierárquicos dentro de uma indústria, reduz-se o custo de fabricação, pela eficiência da manipulação do produto, aumenta-se a produtividade e se estabelece um novo conceito em automação industrial, a integrabilidade de seus componentes nos mais diferentes níveis como pode ser visto na figura a seguir.

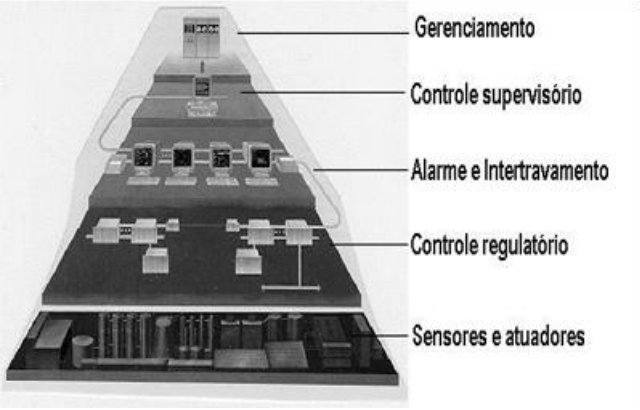

Figura 1.6 - Integração de um sistema industrial

Vistos quais os principais conceitos que são utilizados na automação industrial, bem como os níveis hierárquicos de um sistema industrial, o próximo item dará uma introdução ao controlador lógico programável e seu contexto na automação industrial.

1.2- CLP NO CONTEXTO DA AUTOMAÇÃO INDUSTRIAL

O controle do tipo discreto, voltado aos processos digitais, teve seu início marcado pela utilização de dispositivos eletromecânicos do tipo a relés e contadores. Com o passar dos anos, os circuitos lógicos tornaram-se mais rápidos, compactos e capazes de receber mais informações de entrada, atuando sobre um maior número de dispositivos de saída. Chegamos, assim, aos microprocessadores (UCPs), responsáveis por receber informações da memória, dos dispositivos de entrada, e a partir dessas informações, desenvolver uma lógica para acionar saídas. Então, até o início da década de 60, a utilização de relés eletromecânicos era praticamente a única opção possível. Com o advento dos dispositivos microprocessados, vieram os Controladores Lógicos Programáveis, onde a forma básica de programação é oriunda da lógica de programação dos diagramas elétricos a relés. Próprio para ambientes industriais, os controladores realizam uma rotina cíclica de operação, o que caracteriza seu princípio de funcionamento, e operam apenas variáveis digitais, efetuando controle discreto, portanto.

De um modo mais detalhado um CLP é um aparelho digital que usa memória programável para armazenar instruções que implementam funções como: lógica, sequenciamento, temporização, contagem e operações aritméticas, para controlar através de módulos de entrada e saída (digital e analógica) diversos tipos de máquinas e processos. Dessa forma, os CLPs são equipamentos eletrônicos de controle que atuam a partir da filosofia na qual toda a lógica de acionamento pode ser desenvolvida através de software, que

determina ao controlador a sequencia de acionamento a ser desenvolvida. Este tipo de alteração lógica caracteriza um sistema flexível.

Esta flexibilidade faz com que os CLPs tenham algumas vantagens em relação aos sistemas convencionais, tais como:

- ocupam menos espaço;
- requerem menor potência elétrica;
- podem ser reutilizados;
- são programáveis, permitindo alterar os parâmetros de controle;
- têm maior confiabilidade;
- sua manutenção é mais fácil;
- oferecem maior flexibilidade;
- permitem interface de comunicação com outros CLPs e computadores de controle;
- permitem maior rapidez na elaboração do projeto do sistema.

Os CLPs mais atuais trabalham com padrões de protocolo de comunicação para facilitar a interface com equipamentos de outros fabricantes, e também com Sistemas Supervisórios e Redes Internas de comunicação como o protocolo OPC (Object Linked and embedding for Process Control – o qual permite realizar transferências de objetos entre diferentes aplicações) cujo desenvolvimento foi encabeçado pela *Microsoft*, e tende a se tornar o padrão de relacionamento entre os diversos módulos de *software* ou *hardware* de diferentes fabricantes.

Atualmente, encontramos CLPs utilizados na implementação de painéis seqüenciais de intertravamento, controle de malhas, servo-posicionamento, sistemas SCADA (*Supervisory Control and Data Aquisition*), sistemas de controle estatístico de processo, sistema de controle de estações, sistemas de controle de células de manufatura, etc. Os CLPs são encontrados em processos de: empacotamento, engarrafamento, enlatamento, transporte e manuseio de materiais, usinagem, geração de energia; em sistemas de controle predial de ar condicionado, sistemas de segurança, montagem automatizada, linhas de pintura e sistemas de tratamento de água, existentes em indústrias de alimentos, bebidas, automotiva, química, têxtil, plásticos, papel e celulose, farmacêutica e siderúrgica/metalúrgica.

De uma maneira geral, os CLPs tem aplicação na automação de processos discretos (controle ON-OFF) e na automação de processos contínuos (controle de malhas).

2- FUNCIONAMENTO DE UM CLP E SEUS COMPONENTES

2.1- PRINCÍPIO DE FUNCIONAMENTO DE UM CLP

O Controlador Programável, como todo sistema microprocessado, tem seu princípio de funcionamento baseado nos seguintes passos:

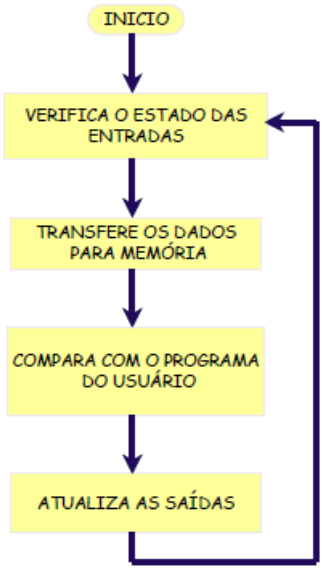
INICIO
VERIFICA O ESTADO DAS ENTRADAS
TRANSFERE OS DADOS PARA MEMÓRIA
COMPARA COM O PROGRAMA DO USUÁRIO
ATUALIZA AS SAÍDAS

Figura 2.1 - Ciclo de varredura de um CLP

- **Inicialização:** No momento em que é ligado o CLP executa uma série de operações pré-programadas, gravadas em seu Programa Monitor:

  - Verifica o funcionamento eletrônico da C.P.U. , memórias e circuitos auxiliares;

  - Verifica a configuração interna e compara com os circuitos instalados;

  - Verifica o estado das chaves principais (RUN/STOP, PROG, etc);

  - Desativa todas as saídas;

  - Verifica a existência de um programa de usuário;

  - Emite um aviso de erro caso algum dos itens acima falhe.

- **Verificar estado das entradas:** O CLP lê o estado de cada uma das entradas, verificando se alguma foi acionada. O processo de leitura recebe o nome de Ciclo de Varredura (*Scan*) e normalmente é de alguns microssegundos (*scan time*).
- **Tranferir os dados para memória:** Após o Ciclo de Varredura, o CLP armazena os resultados obtidos em uma região de memória chamada de Memória Imagem das

Entradas e Saídas. Ela recebe este nome por ser um espelho do estado das entradas e saídas. Esta memória será consultada pelo CLP no decorrer do processamento do programa do usuário.

- **Comparar com o programa do usuário:** O CLP ao executar o programa do usuário, após consultar a Memória Imagem das Entradas, atualiza o estado da Memória Imagem das Saídas, de acordo com as instruções definidas pelo usuário em seu programa.
- **Atualizar as saídas:** O CLP escreve o valor contido na Memória das Saídas, atualizando as interfaces ou módulos de saída. Inicia - se então, um novo ciclo de varredura.

2.2- COMPONENTES DE UM CLP

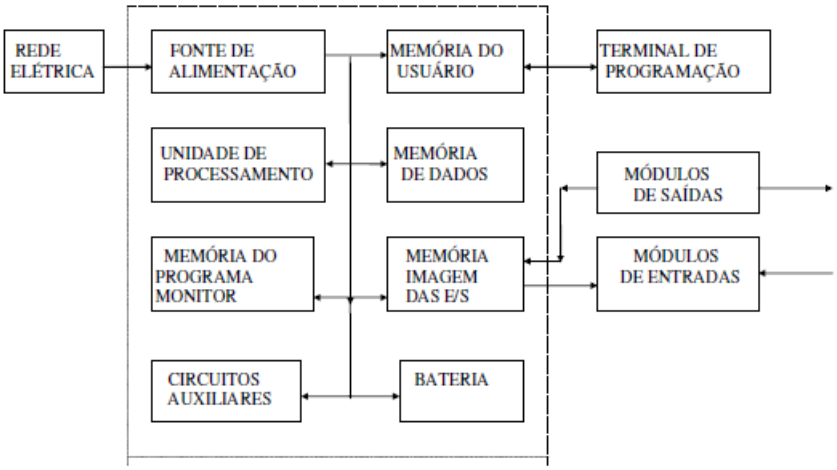

Figura 2.2 - Componentes de um CLP

Fonte de alimentação: A fonte de alimentação do CLP converte a tensão alternada da rede elétrica para uma tensão contínua, ou seja, converte a tensão da rede de 110 ou 220 VCA em +5VCC, +12VCC ou +24VCC para alimentar os circuitos eletrônicos, as entradas e as saídas.

Unidade de processamento: Também chamada de CPU, é responsável pelo funcionamento lógico de todos os circuitos. Nos CLPs modulares a CPU está em uma placa (ou módulo) separada das demais, podendo-se achar combinações de CPU e Fonte de Alimentação. Nos CLPs de menor porte a CPU e os demais circuitos estão todos em único módulo. As características mais comuns são:

- Microprocessadores ou Microcontroladores de 8 ou 16 bits (INTEL 80xx, MOTOROLA 68xx, ZILOG Z80xx, PIC 16xx);

- Endereçamento de memória de até 1 Mega Byte;

- Velocidades de CLOCK variando de 4 a 30 MHZ;

- Manipulação de dados decimais, octais e hexadecimais.

Bateria: As baterias são usadas nos CLPs para manter o circuito do Relógio em Tempo Real, reter parâmetros ou programas (em memórias do tipo RAM) ,mesmo em caso de corte de energia, guardar configurações de equipamentos etc. Normalmente são utilizadas baterias recarregáveis do tipo Ni-Ca ou Li. Nestes casos, incorporam se circuitos carregadores.

Memória do programa supervisor: O programa supervisor é responsável pelo gerenciamento de todas as atividades do CLP. Não pode ser modificado pelo usuário e fica normalmente em memórias do tipo PROM, EPROM, EEPROM. Ele funciona de maneira similar ao Sistema Operacional dos microcomputadores. É o Programa Monitor que permite a transferência de programas entre um microcomputador ou Terminal de Programação e o CLP, gerenciar o estado da bateria do sistema, controlar os diversos opcionais etc.

Memória do usuário: É a memória que armazena o programa do usuário. Esta memória, reservada ao programa do usuário, contém alguns Kbytes de palavras-livres que serão processadas pela CPU. A cada ciclo, a CPU processa este programa, atualiza a memória de dados internos e as imagens das entradas e saídas. Constituída por memórias do tipo RAM, EEPROM ou FLASH-EPROM.

Memória de dados: É a região de memória destinada a armazenar os dados do programa do usuário. Estes dados são valores de temporizadores, valores de contadores, códigos de erro, senhas de acesso, etc. São normalmente partes da memória RAM do CLP. São valores armazenados que serão consultados e ou alterados durante a execução do programa do usuário. Em alguns CLPs, utiliza-se a bateria para reter os valores desta memória no caso de uma queda de energia.

Memória imagem das entradas/saídas: Sempre que a CPU executa um ciclo de leitura das entradas ou executa uma modificação nas saídas, ela armazena o estado da cada uma das entradas ou saídas em uma região de memória denominada Memória Imagem das Entradas/Saídas. Essa região de memória funciona como uma espécie de “tabela” onde a CPU irá obter informações das entradas ou saídas para tomar as decisões durante o processamento do programa do usuário.

Circuitos auxiliares: São circuitos responsáveis para atuar em casos de falha do CLP. Alguns deles são:

- POWER ON RESET: desliga todas as saídas assim que o equipamento é ligado, isso evita que possíveis danos venham a acontecer.

- POWER DOWN: monitora a tensão de alimentação salvando o conteúdo das memórias antes que alguma queda de energia possa acontecer.

- WATCH DOG TIMER: o "cão de guarda" deve ser acionado em intervalos periódicos, isso evita que o programa entre em "loop" no caso de falha do microprocessador.

Módulos de entradas e saídas: Os módulos de entradas e saídas realizam a conexão física entre a CPU e o mundo externo através de circuitos de interfaceamento. Os módulos de entrada têm que ser seguros contra destruição das entradas por excesso ou alimentação de tensão indevida; e devem possuir filtros de supressão para impulsos parasitórios.

Figura 2.3 - Esquema de um módulo de entrada de um CLP

Os módulos de saídas devem ser amplificados e possuir proteção contra curto-circuito.

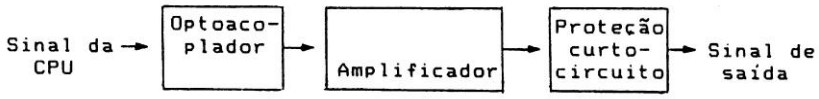

Figura 2.4 - Esquema de um módulo de saída de um CLP

Os módulos de entradas e saídas podem ser subdivididos em dois grupos distintos: digitais (discretos) e analógicos (numéricos).

- **Como as entradas analógicas são lidas:** A tensão ou corrente de entrada é convertida para um código digital proporcional ao valor analógico, através de um conversor analógico/digital.

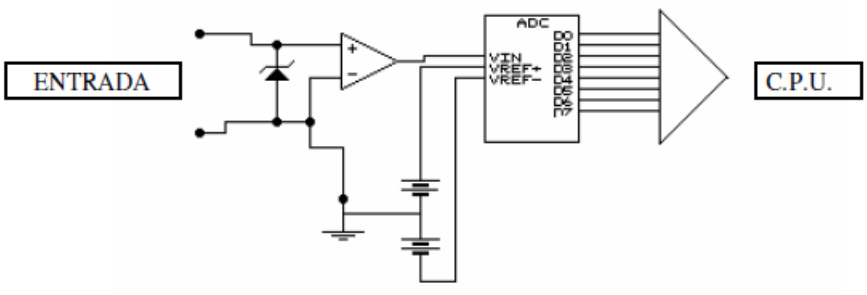

Figura 2.5 - Exemplo de um circuito de entrada analógica

- **Como são tratadas as saídas analógicas:** A interface das saídas analógicas recebe do processador dados numéricos que são convertidos em valores proporcionais de corrente ou tensão e aplicados nos dispositivos de campo.

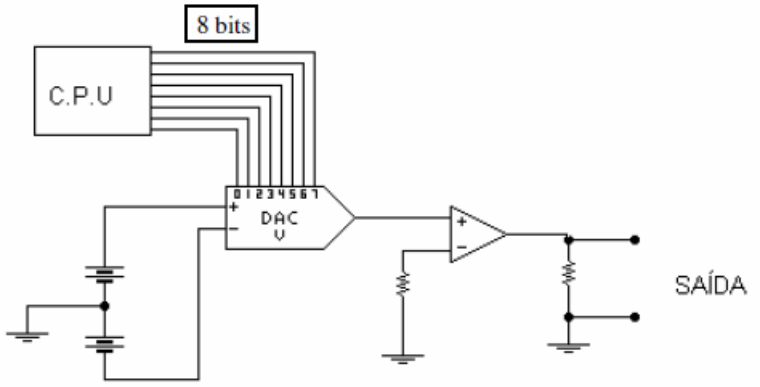

Figura 2.6 - Exemplo de um circuito de saída analógica

3- SFC (GRAFCET)

3.1- CONCEITOS IMPORTANTES

O Grafcet, também conhecido como SFC (Sequential Functional Charts), é um método gráfico de apoio à concepção de sistemas industriais automatizados, que permite representar, através de modelos do tipo dos gráficos de estados, o comportamento de sistemas sequenciais.

A partir do modelo Grafcet de um sistema, pode ser gerado automaticamente o programa do controlador desse sistema, sendo muito mais simples construir o modelo Grafcet, do que desenvolver o programa do controlador.

Na fase de representação de sistemas automatizados, o Grafcet oferece a possibilidade de:

- Descrever o funcionamento de sistemas complexos através de modelos compactos e, dessa forma, estruturar a representação desses sistemas;
- Com base nos modelos, simular o funcionamento dos sistemas e, assim, detectar e eliminar eventuais erros de representação antes de passar à fase de implementação;
- A partir do modelo do sistema, gerar automaticamente o programa do autômato programável.

Os elementos de um Grafcet são: etapas, transições, arcos, receptividade, ações e regras de evolução, como podem ser observados no exemplo a seguir.

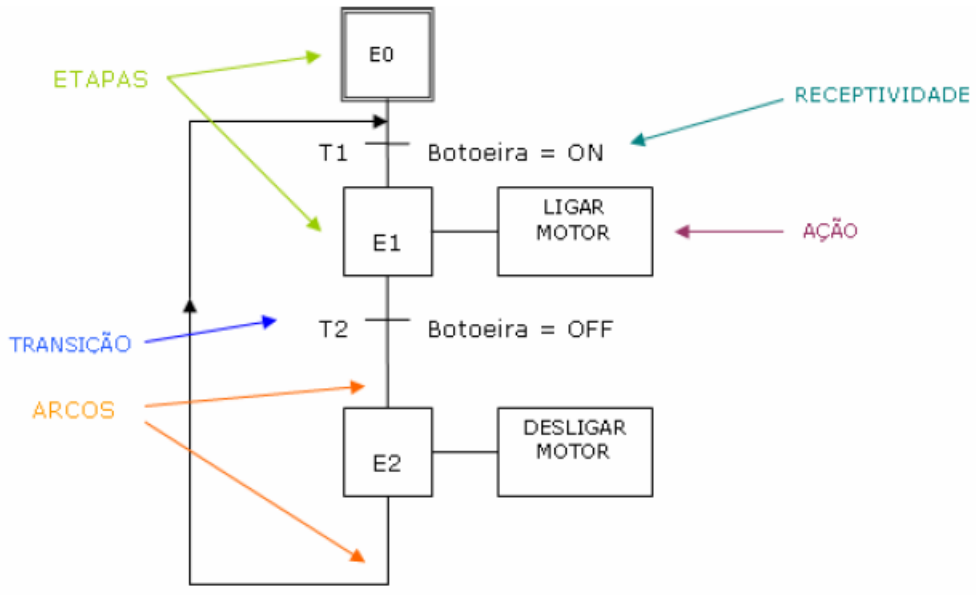

Figura 2.1 - Exemplo de representação em Grafcet

O Grafcet descreve a sequência em que as ações serão executadas de acordo com sinais de entrada. Cada ação está associada a uma etapa do Grafcet e somente é executada quando a etapa estiver ativa. As transições controlam a evolução do Grafcet, ou seja, a desativação e a ativação das etapas. Cada transição está associada a uma receptividade, ou expressão lógica das variáveis de entrada, que quando verdadeira, causará a desativação da etapa anterior à transição e ativação da etapa posterior. A ligação entre etapas e transições é realizada por arcos orientados que determinam o sentido de evolução do Grafcet.

3.2- DESCRIÇÃO DOS SEUS ELEMENTOS

3.2.1- ETAPAS

Uma etapa é um estado no qual o comportamento do circuito de comando não se altera frente a entradas e saídas, ou seja, cada etapa corresponde a uma condição invariável e bem definida do sistema descrito.

Em um determinado instante uma etapa pode estar ativa ou inativa. O conjunto de etapas ativas num determinado instante determina a situação em que se encontra o Grafcet.

Figura 3.2 - Indicação de uma etapa ativa

A Etapa inicial é a etapa que se torna ativa logo após início do funcionamento do Grafcet.

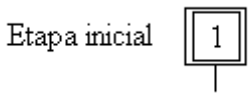

Figura 3.3 - Bloco que representa uma etapa inicial

3.2.2- ARCOS ORIENTADOS

Os arcos orientados indicam a sequência do Grafcet pela interligação de uma etapa a uma transição e desta a outra etapa. O sentido convencionado é de cima para baixo, quando não for o caso, deve-se indicá-lo.

3.2.3- AÇÕES

As ações representam os efeitos que devem ser obtidos sobre os mecanismos controlados em uma determinada situação (“o que deve ser feito”) e são executadas quando a etapa associada estiver ativa. Representam também ordens de comando (“como deve ser feito”).

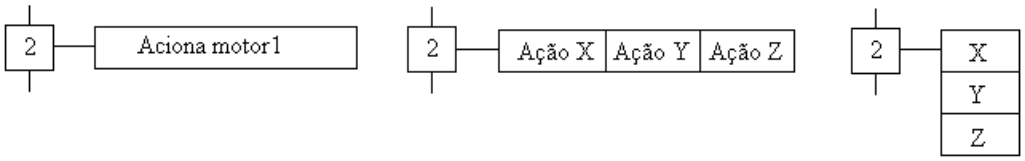

Figura 3.4 - Representação das ações em Grafcet

É preciso definir se a ação será mantida ou finalizada após a desativação da etapa.

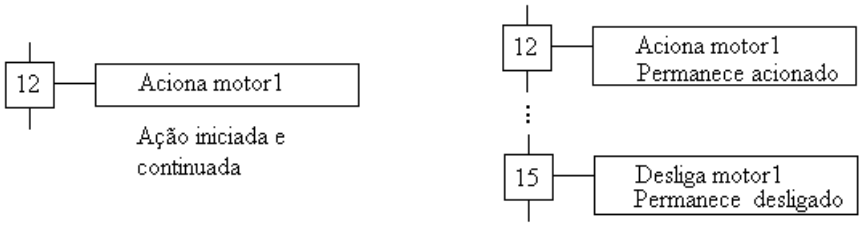

Figura 3.5 - Prosseguimento de ações definidas em Grafcet

Quando quisermos condicionar e/ou limitar uma ação.

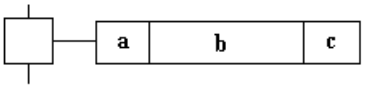

Figura 3.6 - Campos das ações em Grafcet

- A largura do campo **b** é maior que o dobro da de **a** ou **b**.
- Campo ‘a’ – deve conter o qualificador que define como a ação associada à etapa será executada.
- Campo ‘b’ – declaração textual ou simbólica da ação.
- Campo ‘c’ – referência do sinal de retorno que será verificado pela transição seguinte.

Os qualificadores definidos são: S (stored), D (delayed), L (time limited), P (pulse shaped) e C (condition).

- **S** (armazenada/mantida)

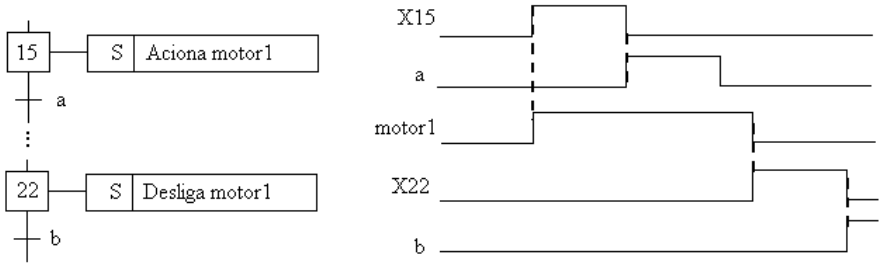

Figura 3.7 - Exemplo em Grafcet utilizando qualificador de armazenamento

- **D** (atrasada)

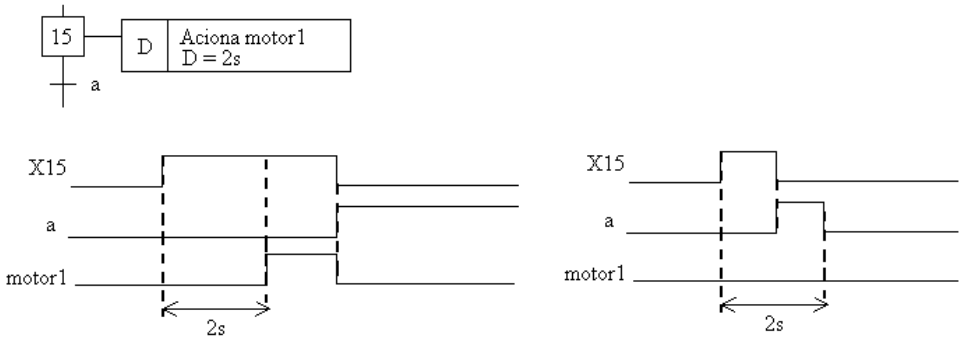

Figura 3.8 - Exemplo em Grafcet utilizando qualificador de atraso

- **L** (limitada por tempo)

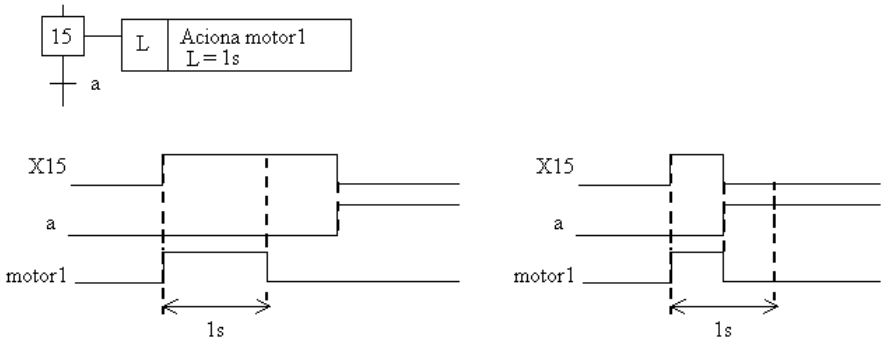

Figura 3.9 - Exemplo em Grafcet utilizando qualificador de tempo limitado

- **P** (pulsada/tempo muito curto)

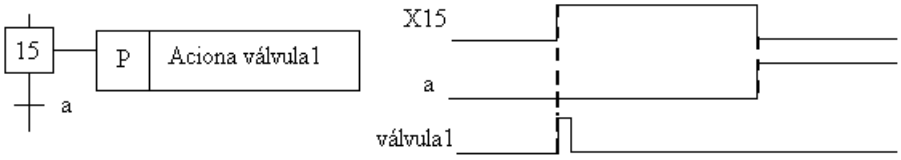

Figura 3.10 - Exemplo em Grafcet utilizando qualificador de forma de pulso

- **C** (condicional)

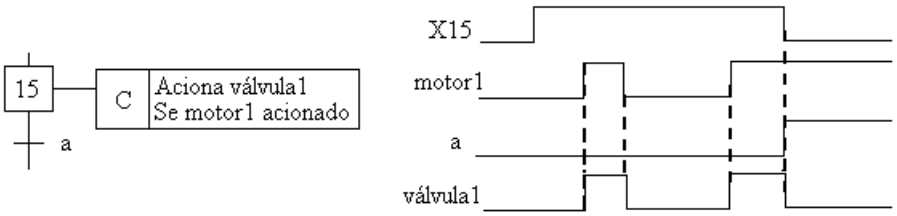

Figura 3.11 - Exemplo em Grafcet utilizando qualificador de condição

Além de um único qualificador, uma ação pode ser detalhada por meio de uma combinação de qualificadores.

- SD – ação é armazenada e iniciada após tempo, mesmo que a etapa não esteja mais ativa.
- DS – ação é iniciada após tempo e continuada até ser finalizada por uma etapa seguinte.
- CSL – ação é iniciada desde que a condição lógica seja satisfeita e mantida por tempo especificado.

Quando a etapa não tem ação associada, mas tem um estado associado.

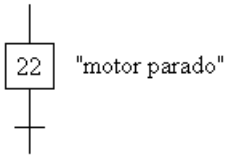

Figura 3.12 - Exemplo de etapa sem ação associada

3.2.4- TRANSIÇÕES

Uma transição representa uma evolução possível entre dois dos estados do sistema e é representada graficamente por traços nos arcos orientados que ligam etapas e significam a evolução do Grafcet de uma situação para outra.

Em um dado instante, uma transição pode está válida ou não. Uma transição está válida quando todas as etapas imediatamente precedentes estiverem ativas.

A passagem de uma situação para outra só é possível com a validade de uma transição, e se dá com a ocorrência da transição.

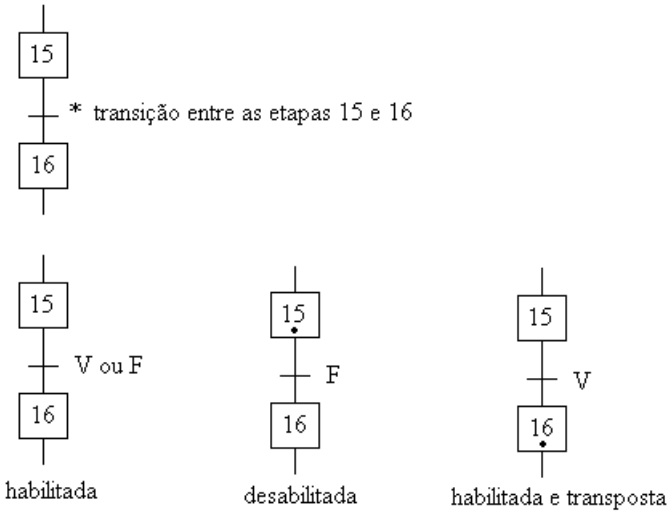

Figura 3.13 - Procedimento das transições em Grafcet

3.2.5- RECEPTIVIDADE

É a função lógica combinacional associada a cada transição. Quando em estado lógico verdadeiro, irá habilitar a ocorrência de uma transição válida.

Uma receptividade é associada a:

- Variáveis lógicas oriundas de sinais de entrada do sistema;
- Variáveis internas de controle;
- Resultado da comparação de contadores e temporizadores;
- Informação do estado de outra etapa;
- Condicionada a uma determinada situação do Grafcet;

Uma receptividade também pode estar associada ao sentido de comutação de uma variável lógica, seja pela borda de subida, seja pela borda de descida. Por exemplo:

↑A (borda de subida da variável A)

↓A (borda de descida da variável A)

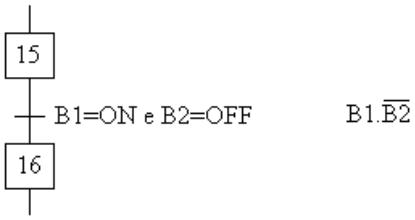

Figura 3.14 - Exemplo de receptividade em Grafcet

- Condições detalhadas:

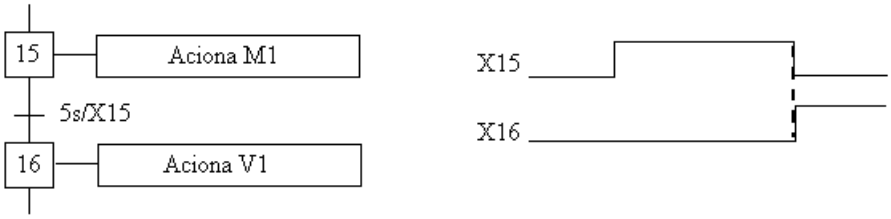

Figura 3.15 – Exemplo com condições detalhadas em Grafcet

- Transição incondicional:

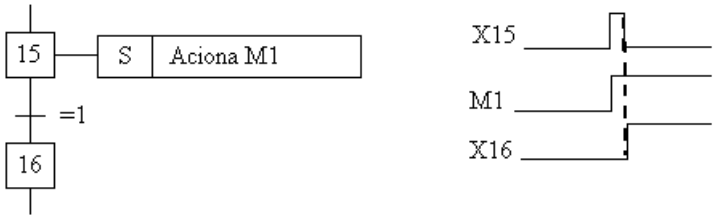

Figura 3.16 - Exemplo com transição incondicional em Grafcet

3.3- REGRAS DE EVOLUÇÃO

Antes de qualquer coisa, qualquer que seja a sequência percorrida em um Grafcet, deve existir sempre uma alternância entre etapas e transições, isto é:

- duas etapas nunca podem estar ligadas diretamente, mas devem estar separadas por uma transição;
- duas transições nunca podem estar ligadas diretamente, mas devem estar separadas por uma etapa.

Sendo assim, não são permitidas as seguintes estruturas:

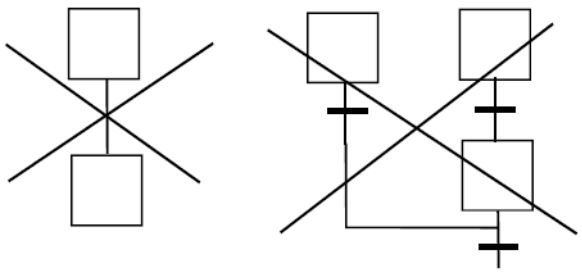

Figura 3.17 - Exemplos de estruturas não permitidas em Grafcet

A evolução de um Grafcet de uma situação a outra corresponde à ocorrência de uma transição. A ocorrência de uma transição possui tempo de duração impulsional (um ciclo de varredura no CLP). Temos então, as seguintes regras de evolução em Grafcet:

1) Situação inicial: deve existir ao menos uma etapa inicial. É ativada incondicionalmente.

2) Transposição de uma transição: uma transição só é transposta se:

   - Estiver habilitada (todas as etapas precedentes ativas);
   - A condição associada for verdadeira.

3) Evolução das etapas ativas: na ocorrência de uma transição, ocorre a desativação de todas as etapas imediatamente precedentes, e a ativação de todas as etapas imediatamente seguintes.

4) Transposição simultânea de transições:

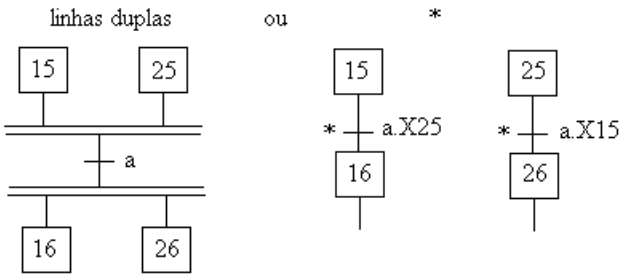

Figura 3.18 - Transposição simultânea de transições em Grafcet

5) Condições verdadeiras e imediatamente seguintes: se, no instante de ativação de uma etapa, a condição de transição desta à etapa seguinte for verdadeira, ela não ocorrerá.

6) Ativação e desativação simultâneas de uma etapa: a ativação é prioritária em relação à desativação.

7) Tempo nulo: o tempo para a transposição de uma transição ou ativação de uma etapa é nulo.

O exemplo abaixo demonstra o procedimento de evolução em Grafcet:

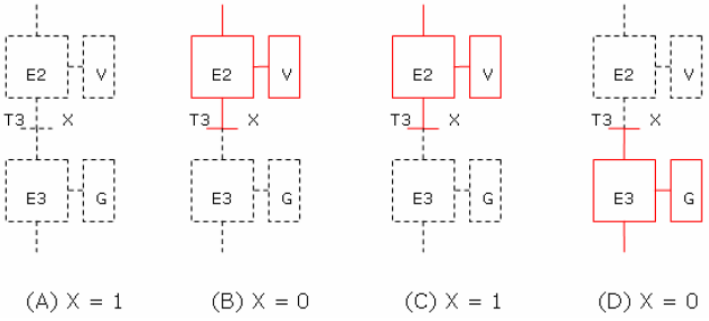

Figura 3.19 - Exemplo de evolução em Grafcet

No instante A embora a receptividade X seja verdadeira (X = 1), a etapa anterior à transição T3 não está ativa, dessa forma não ocorre a ativação da etapa E3. No instante B a etapa E2 esta ativa, mas a receptividade X é falsa (X = 0), dessa forma a etapa E2 permanecerá ativa até que a receptividade X associada a transição T3 torne-se verdadeira, o que ocorre no instante C. Isto causa uma evolução do Grafcet para o estado apresentado no instante D, onde a etapa E2 é desativada e E3 é ativada.

3.4- ESTRUTURA SEQUENCIAL

3.4.1- SEQUENCIA ÚNICA

Uma cadeia de etapas e transições dispostas de forma linear onde uma etapa é seguida de apenas uma transição, e uma transição é seguida de apenas uma etapa.

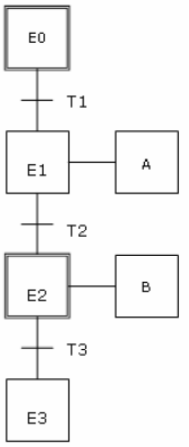

Figura 3.20 - Exemplo de sequencia única em Grafcet

3.4.2- SELEÇÃO ENTRE SEQUENCIAS

Ocorrência de situações em que uma determinada sequência deve ser executada no lugar de outras.

- **Divergência seletiva** ou **divergência em OU**: uma divergência seletiva é precedida por uma etapa e sucedida por sequências iniciadas por transições. É recomendável que as transições numa divergência seletiva sejam receptivas e possuam condições lógicas mutuamente exclusivas entre si.

Regra de interpretação do Grafcet: a sequência situada mais à esquerda terá prioridade de execução.

- **Convergência seletiva** ou **convergência em OU**: retorno do Grafcet a uma estrutura linear. Uma convergência seletiva é sucedida por uma etapa e precedida por sequências finalizadas por transições.

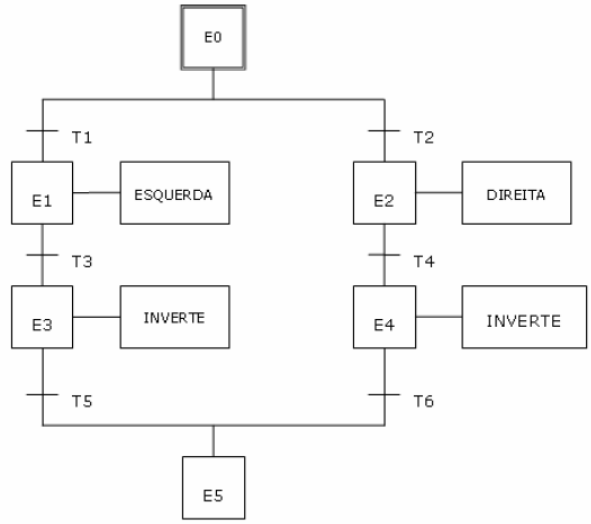

Figura 3.21 - Exemplo de seleção entre sequencias em Grafcet

3.4.3- CASOS PARTICULARES DE SELEÇÃO ENTRE SEQUENCIAS

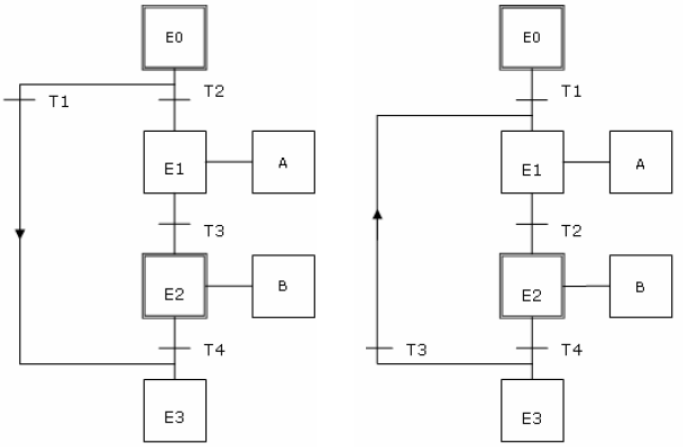

Figura 3.22 - Exemplo de salto em sequencia e repetição em sequencia em Grafcet

3.4.4- SEQUÊNCIAS PARALELAS

Ocorre quando duas ou mais sequências devem ser executadas ao mesmo tempo.

- **Divergência simultânea** ou **divergência em E**: uma divergência simultânea é precedida por uma transição e sucedida por sequências iniciadas por etapas.

- **Convergência simultânea** ou **convergência em E**: retorno do Grafcet a uma estrutura linear. Uma convergência simultânea é sucedida por uma transição e precedida por sequências terminadas por etapas. O paralelismo só é encerrado quando todas as suas sequências estiverem concluídas.

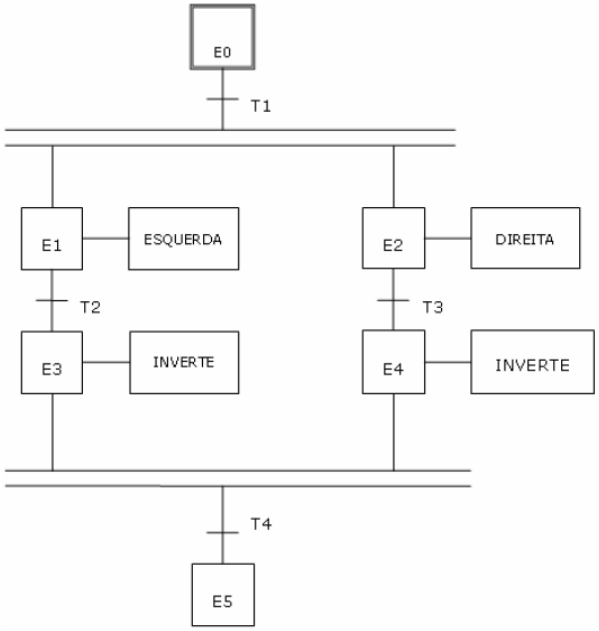

Figura 3.23 - Exemplo de sequência paralela em Grafcet

3.5- EXEMPLOS UTILIZANDO GRAFCET

3.5.1- COMANDO DE UMA LÂMPADA POR UMA CHAVE ÚNICA

Neste exemplo, o Grafcet desenvolvido representa o comando de uma lâmpada por uma chave liga/desliga.

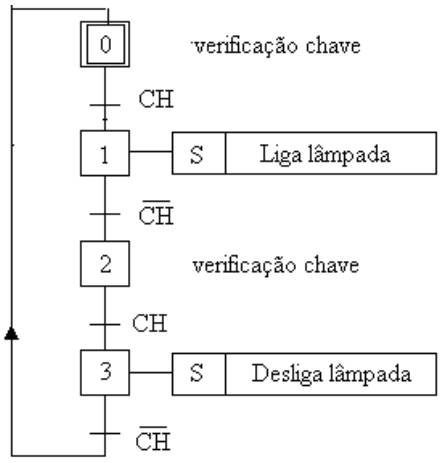

Figura 3.24 - Grafcet referente ao comando de uma lâmpada por uma chave única

3.5.2- LIGAR UMA LÂMPADA DURANTE UM TEMPO

Neste exemplo, o Ladder desenvolvido representa uma lâmpada que ao ser ligada, permanece assim por 5s antes de ser desligada novamente.

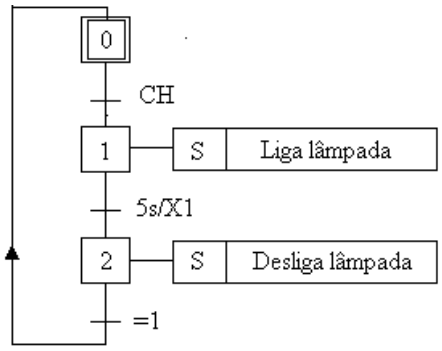

Figura 3.25 - Grafcet referente ao comando de ligar uma lâmpada durante um certo tempo

3.5.3- SISTEMA CARREGAMENTO DE UM CARRO

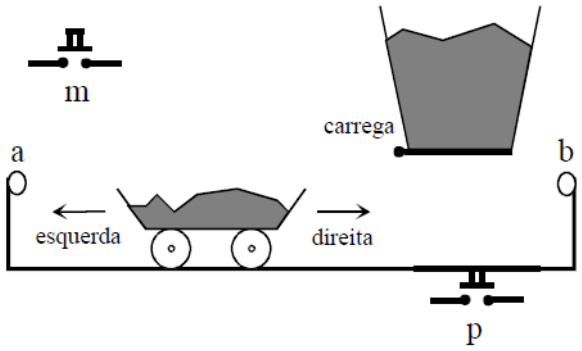

Figura 3.26 – Esquema do sistema de carregamento de um carro

Pressionando o botão M, o carro desloca-se para a direita até atingir o fim de curso "b", quando então inicia o carregamento até atingir o peso determinado pelo sensor "p". Neste caso, a vávula deve ser fechada e o carro deve retornar a posição inicial. Esta é detectada pelo fim de curso "a". O movimento para a direita é realizado pelo motor M1 e para a esquerda pelo motor M2.

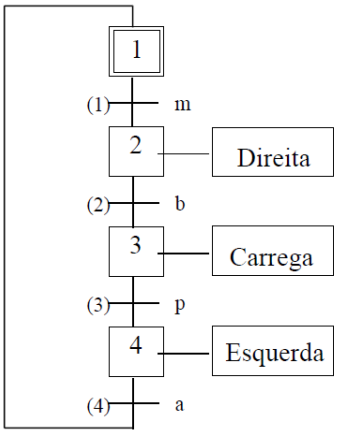

Figura 3.27 - Grafcet referente ao sistema de carregamento de um carro

3.5.4- SISTEMA DE PARTIDA ESTRELA/TRIÂNGULO

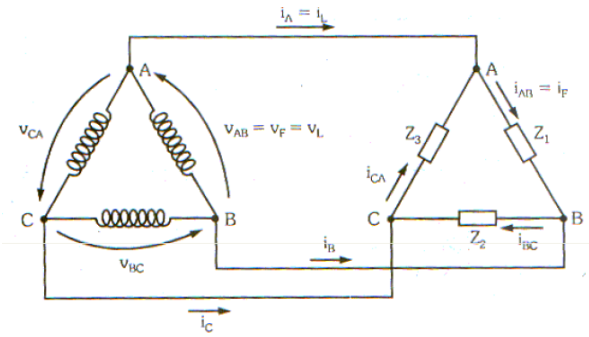

Figura 3.28 - Esquema de um sistema de partida estrela/triângulo

Neste exemplo, um motor trifásico deve ser acionado por meio de partida estrela/triângulo. Para isto são utilizados três contatores (acionamento geral e chaveamento estrela e triângulo) e uma botoeira. O sistema automatizado deve apresentar o seguinte comportamento:

- ao ser acionada a botoeira pela primeira vez, os contatores de acionamento geral e chaveamento estrela são acionados;
- Dois após, apenas o contator de chaveamento estrela é desligado;
- ao ser confirmado o desligamento do contator de chaveamento estrela, o contator de chaveamento triângulo é acionado;
- os contatores de acionamento geral e chaveamento triângulo permanecem acionados;
- a qualquer instante, um novo acionamento da botoeira ocasiona o desligamento dos contatores acionados, retornando à condição inicial.

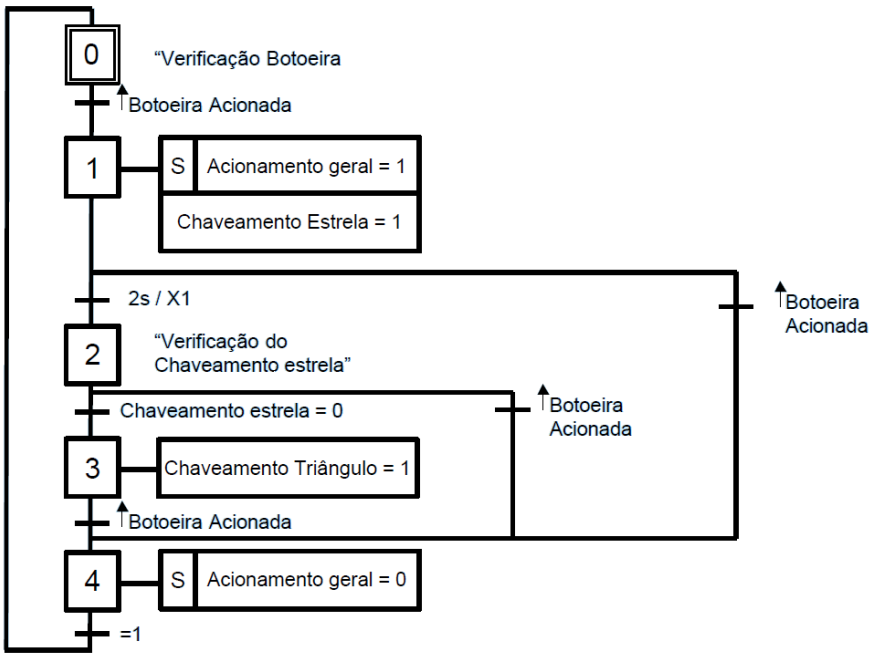

Figura 3.29 - Grafcet referente ao sistema de partida estrela/triângulo

3.5.5- SISTEMA DE UM PORTÃO AUTOMÁTICO

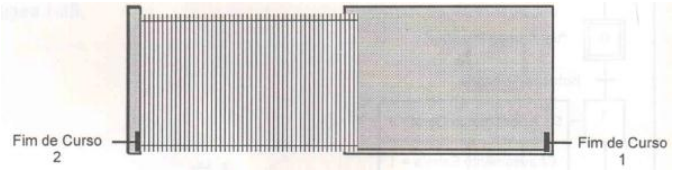

Figura 3.30 - Esquema do sistema de um portão automático

O acionamento de um portão deve ser realizado automaticamente por intermédio de uma única botoeira, que serve para abrir, fechar e interromper a abertura e fechamento do portão. Acoplado mecanicamente ao portão um motor elétrico realiza os movimentos pela inversão do sentido de rotação. O sistema automatizado deverá respeitar as condições abaixo:

- No primeiro acionamento da botoeira, inicia-se a abertura do portão;
- A parada da abertura se dá por meio de novo acionamento da botoeira (com o portão em movimento) ou pela abertura total do portão (chave de fim de curso 1);
- Estando o portão totalmente aberto (fim de curso 1 acionado), ou tendo sido interrompida a abertura, no próximo acionamento da botoeira, inicia-se o fechamento;
- A parada do fechamento se dá por meio de novo acionamento da botoeira (com o portão em movimento) ou pelo fechamento total do portão (chave de fim de curso 2);
- Estando o portão totalmente fechado (fim de curso 2 acionado), ou tendo sido interrompido o fechamento, no próximo acionamento da botoeira, inicia-se a abertura.

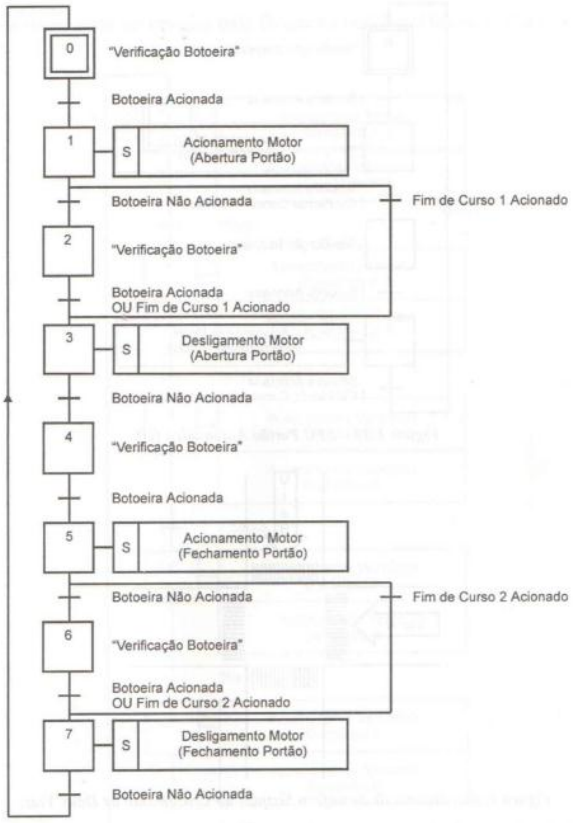

Figura 3.31 - Grafcet referente ao sistema de um portão automático

3.5.6- MÁQUINA DE FAZER BEBIDAS QUENTES

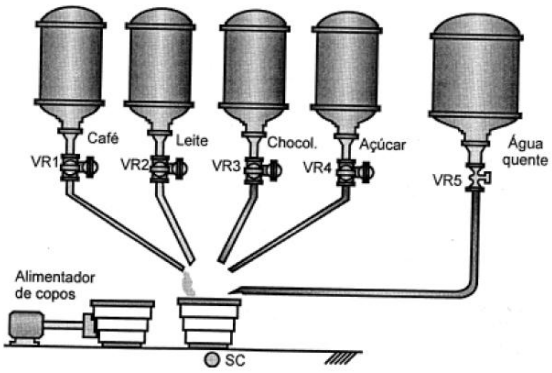

Figura 3.32 - Esquema de uma máquina de fazer bebidas quentes

- O processo a ser automatizado é uma máquina dispensadora de bebidas quentes que pode fornecer as seguintes opções ao usuário: B1-café puro, B2-café com leite e B3-chocolate quente;
- O sistema é dotado de cinco reservatórios: R1-café solúvel, R2-leite em pó, R3-chocolate, R4-açúcar e R5-água quente. A dosagem de cada produto no copo descartável é feita pela abertura temporizada de válvulas VR1, VR2, VR3, VR4 e VR5, respectivamente. Há também um dispositivo eletromecânico (AC) para a alimentação do copo descartável, o qual posiciona corretamente apenas um copo a cada vez que for atuado;

- O sistema prevê ainda três níveis de liberação de açúcar: A1-amargo, A2-doce, A3-extra-doce;
- Como condição inicial de funcionamento, um copo deve ser posicionado corretamente, o qual é monitorado pelo sensor SC.
- Como condição de finalização, o copo cheio deve ser retirado;
- Assim, com a condição inicial satisfeita, um nível de açúcar e um tipo de bebida pré-selecionados, com o pressionar da botoeira de partida (BP) inicia-se o processo de preparo pela abertura temporizada das eletroválvulas;
- Primeiro ocorre a liberação do açúcar com os tempos de abertura de VR4 de 4 segundos para doce, 6 segundos para extra-doce e sem liberação para amargo. Após isto, inicia-se então o preparo de uma das seguintes receitas:
    - **Café puro**: 3 segundos de café e 5 segundos de água quente;
    - **Café com leite**: 2 segundos de café, 3 segundos de leite e 7 segundos de água quente;
    - **Chocolate**: 2 segundos de leite, 3 segundos de chocolate e 6 segundos de água quente.

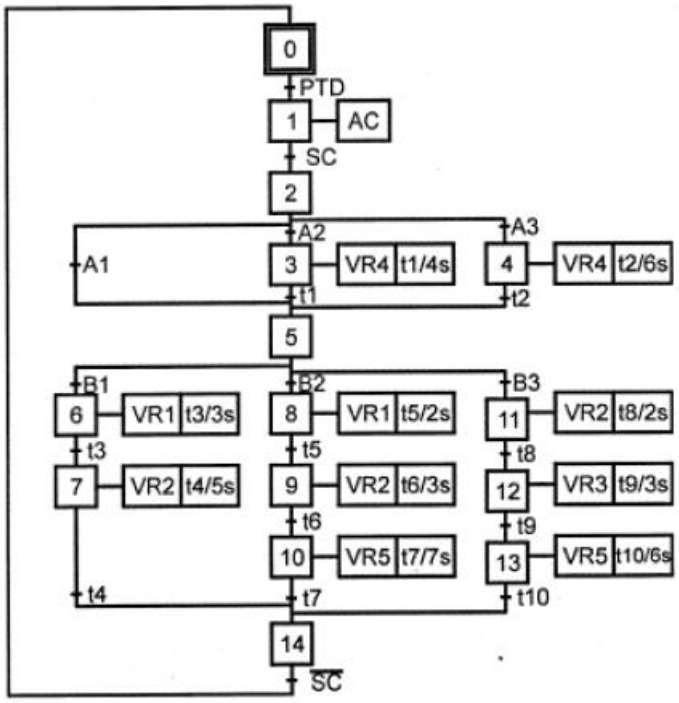

Figura 3.33 - Grafcet referente à máquina de fazer bebidas quentes

4- LADDER (HI TECNOLOGIA)

4.1- INTRODUÇÃO AO LADDER

A programação traduz as funções a serem executadas; para tanto ela deve ser o mais simples possível. Utilizando-se de linguagem específica, baseando-se na memotécnica, a linguagem de programação usa abreviações, figuras e números de tal forma a formar-se acessível a todos os níveis tecnológicos.

Os tipos de funções são associações lógicas (“E”, “OU”, etc), funções de memória (SET, RESET, etc), funções de contagem, temporização, aritméticas e outras mais específicas. A forma visual que a instrução se apresenta depende unicamente do tipo de sistema utilizado pelo programador. Seja por exemplo, a associação lógica “OU” entre duas informações que chamaremos de entradas por traduzirem informações do processo. O resultado desta associação será armazenado em uma memória para depois ser utilizado, na dependência da ordem de operação. Podemos representar essa associação na forma de diagrama de contatos (Ladder).

A linguagem Ladder é uma representação gráfica da linguagem de programação do CLP. Também conhecida como lógica de diagrama de contatos. A linguagem Ladder é o sistema de representação que mais se assemelha à tradicional notação de diagramas elétricos. Verifique por exemplo o esquema conceitual de um sistema CLP, mostrado na figura abaixo para representar o seu funcionamento juntamente com um circuito elétrico simples. As entradas físicas reais estão fixadas a um módulo de entrada (esquerda) enquanto as saídas estão fixadas a um módulo de saída (direita). No centro, vê-se a representação lógica que a CPU deve processar na linguagem Ladder. Neste caso, se o Input 1 for fechado, a Output 1 (campainha) é ligada.

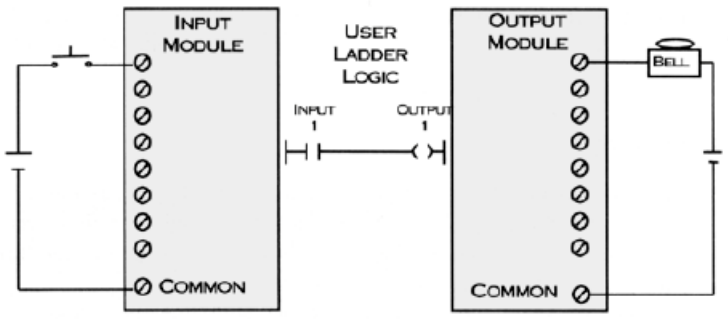

Figura 4.1 - Esquema do funcionamento de um CLP com um circuito elétrico simples

O exemplo a seguir ilustra representações de um circuito em série simples composto por dois interruptores e uma lâmpada em diversas maneiras:

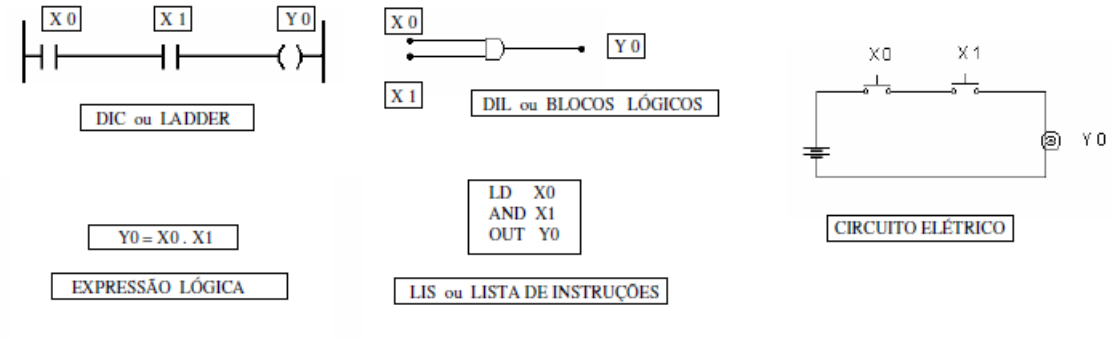

Figura 4.2 - Diversas representações de um circuito em série simples com 2 interruptores e 1 lâmpada

O nome Ladder deve-se à representação da linguagem se parecer com uma escada (ladder), na qual duas barras verticais paralelas são interligadas pela Lógica de Controle, formando os degraus (rungs) da escada. Portanto, a cada Lógica de Controle existente no Programa de Aplicação dá-se o nome de rung, a qual é composta por Colunas e Linhas, conforme apresentado na figura a seguir.

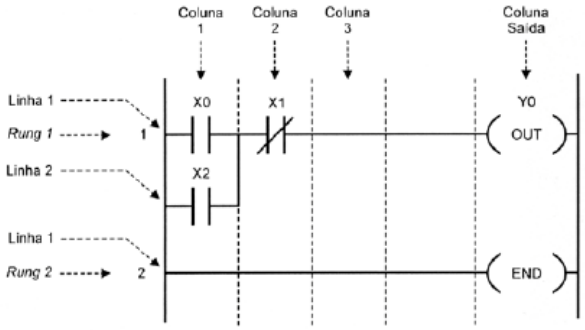

Figura4.3 - Estrutura da linguagem Ladder

A quantidade de Colunas e Linhas, ou Elementos e Associações, que cada rung pode conter é determinada pelo fabricante do CLP, podendo variar conforme a CPU utilizada. Em geral, este limite não representa uma preocupação ao usuário durante o desenvolvimento do Programa de Aplicação, pois os Softwares de Programação indicam se tal quantidade foi ultrapassada, por meio de erro durante a compilação do Programa de Aplicação.

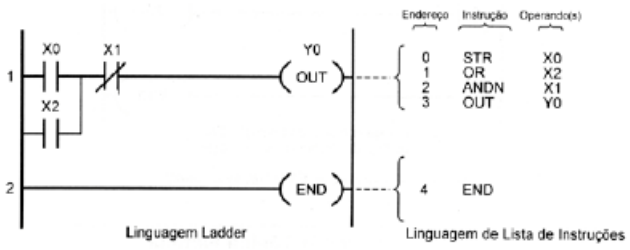

Figura 4.4 – Endereçamento de instruções durante a execução de um programa em Ladder

A relação entre o símbolo gráfico da Linguagem Ladder e a Instrução a ser executada pode ser verificada nos Endereços 0 e 1 do Programa em Linguagem de Lista de Instruções. Neste caso, a representação em Linguagem Ladder para os Elementos XO e X2 são Contatos Normalmente Abertos idênticos. Porém, a localização de cada um na Lógica de Controle determina Instruções diferentes, ou seja, o Contato Normalmente Aberto de XO, por iniciar o rung, determina a Instrução 'Store' (STR XO) e o Contato Normalmente Aberto de X2 (com representação gráfica idêntica à de XO), por estar em paralelo com XO, determina a Instrução 'Or' (OR X2). Esta característica da Linguagem Ladder normalmente facilita o desenvolvimento do Programa de Aplicação, uma vez que o usuário precisa certificar-se apenas se a associação desejada é aceita pela CPU utilizada, não se prendendo à Instrução propriamente dita.

4.2- INTRODUÇÃO AO AMBIENTE DE PROGRAMAÇÃO

O ambiente de desenvolvimento utilizado para programar este CLP da HI Tecnologia (SPDSW) é composto por um Menu Principal com as seguintes opções:

- **Projeto**: carrega um projeto existente ou cria um novo;
- **Programa**: edita um programa em Ladder;
- **Controlador**: gera um código executável para ser enviado ao controlador do CLP;
- **Depurador**: permite a visualização do programa em funcionamento no microcontrolador
- **Supervisão**: gera a supervisão de blocos de controle (PID) ou gráficos de tendência (“Trend Variáveis”)

Os principais operadores utilizados neste CLP são:

- **I**: Representa as entradas digitais. As diferentes entradas são distinguidas através de números sequenciais. Ex: I1, I2, I3, etc.
- **O**: Representa as saídas digitais. As diferentes saídas são distinguidas através de números sequenciais. Ex: O1, O2, O3, etc.
- **R**: Representa um contato auxiliar. Estes não tem conexão direta com o meio físico (processo) e são úteis na definição das lógicas. Também são diferenciados através de números sequenciais. Ex: R1, R2, R3, etc.

- **M**: Memória destinada a guardar valores inteiros de 16 bits, ou seja, números de –32768 a +32768.
- **D**: Memória destinada a armazenar valores reais, ou seja, números de -10^{-38} a $+10^{+38}$.
- **K**: Memória destinada a armazenar uma constante inteira de 16 bits. Pode assumir valores de –999 a +9999.
- **H**: Representa o valor de uma constante Hexadecimal. Pode assumir valores na faixa de 0 a FFFF.
- **Q**: Representa o valor de uma constante real. Pode ser usado com números na faixa de -10^{-38} a $+10^{+38}$.
- **E**: Representa um canal de entrada analógica. Usado pelo bloco de movimentação "MOV", quando se deseja lê o valor de uma entrada analógica do controlador em uma memória.
- **S**: Representa um canal de saída analógica. Usado pelo bloco de movimentação "MOV", quando se deseja movimentar o valor de uma memória a uma saída.
- **T**: Representa um elemento de sinalização de eventos, como por exemplo, para identificar uma mensagem. Identifica uma lógica qualquer, presente em um programa. Representa um malha de controle PID.

4.3- ELEMENTOS DO EDITOR LADDER

4.3.1- INSTRUÇÕES E BLOCOS BÁSICOS

1) Contato Normalmente Aberto (NA):

O objetivo deste elemento é funcionar como um contato normalmente aberto de um relé. Se a bobina do relé estiver desenergizada, ele estará aberto interrompendo o circuito lógico. Se a sua bobina estiver energizada, ele estará fechado, deixando que o circuito se complete através dele.

Figura 4.5 - Representação de contato NA em Ladder

2) Contato Normalmente Fechado (NF):

O objetivo deste elemento é funcionar como um contato normalmente fechado de um relé. Se a bobina do relé estiver desenergizada, ele estará fechado, deixando que o circuito se complete através dele. Se a sua bobina estiver energizada, ele estará aberto, interrompendo o circuito.

Figura 4.6 - Representação de contato NF em Ladder

3) Relé:

O objetivo deste elemento é funcionar como uma bobina de um relé. Se estiver energizada, todos os seus contatos normalmente abertos estarão fechados e os normalmente fechados estarão abertos. Se estiver desenergizada, todos os seus contatos normalmente abertos estarão abertos e os normalmente fechados estarão fechados. Para cada relé somente pode existir uma única bobina e não existe um limite de contatos que um relé pode ter, sendo NA ou NF.

Figura 4.7 - Representação de um relé em Ladder

4) Temporizador:

Este bloco é responsável temporização de eventos dentro de um programa. O bloco é o responsável por temporizar eventos dentro de um programa. Ele opera, basicamente, ativando a saída após a contagem de um tempo pré-programado. Este tempo é especificado no parâmetro P2 e seu valor é dado em múltiplos de 0,01 seg, ou seja, um intervalo de 1 segundo será contabilizado se o valor de P2 for igual a 100.

Este elemento é representado por um bloco funcional com duas entradas (A1 e A2) e uma saída (B1) e utiliza dois operadores (P1 e P2). O operador P1 que aparece na parte superior do bloco guarda o valor corrente do tempo e deve ser obrigatoriamente uma memória (M). O parâmetro P2 contêm o valor total do tempo, e pode ser uma memória (M) ou uma constante (K).

A entrada superior (A1) habilita a temporização ("Habilita") e a entrada (A2) do estado desenergizado para o estado energizado nesta entrada faz com que o valor corrente do temporizador (P1) passe a decrementar até zero (final do temporizador). A entrada inferior

(A2) representa a entrada de temporização ("Temporiza"), assim, para temporizar é necessário que a entrada "Habilita" (A1) esteja energizada e toda vez que a entrada "Temporiza" (A2) transitar de desenergizada para energizada o valor corrente da temporização será decrementado até o valor zero, caso não haja uma intervenção da entrada (A2). A saída (B1) indica se a temporização chegou ao final (Temporização = valor final da temporização).

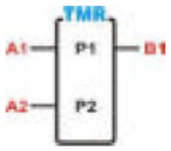

Figura 4.8 - Representação de um temporizador em Ladder

5) Flip-Flop Set:

O objetivo deste elemento é operar como uma entrada *SET* de um "*Flip-Flop*", que é um elemento básico de memória em circuitos elétricos. Este elemento está associado ao controle de saídas digitais (operadores Oxxxx) e contatos auxiliares (operadores Rxxxx) e só pode ser colocado na oitava coluna da janela de edição. Existe uma grande diferença no posicionamento dos elementos de SET e RESET de uma saída digital ou um contato auxiliar no diagrama ladder devido a varredura do Scan.

Figura 4.9 - Representação de um Flip-Flop Set em Ladder

6) Flip-Flop Reset:

O objetivo deste elemento é operar como uma entrada *RESET* de um "*Flip-Flop*", que é um elemento básico de memória em circuitos elétricos. Este elemento está associado ao controle de saídas digitais (operadores Oxxxx) e contatos auxiliares (operadores Rxxxx) e só pode ser colocado na oitava coluna da janela de edição. Existe uma grande diferença no posicionamento dos elementos de SET e RESET de uma saída digital ou um contato auxiliar no diagrama ladder devido a varredura de Scan.

Figura 4.10 - Representação de um Flip-Flop Reset em Ladder

7) Flip-Flop Pulso:

O objetivo deste elemento é ativar a bobina de um relé por um único ciclo de varredura do programa. Esta ativação se dá através de uma transição de subida do sinal aplicado à sua entrada. Este elemento está associado ao controle de saídas digitais (operadores Oxxxx) e contatos auxiliares (operadores Rxxxx) e só pode ser colocado na oitava coluna da janela de edição.

Figura 4.11 - Representação de um Flip-Flop Pulso em Ladder

8) Flip-Flop Oscilador:

O objetivo deste elemento é inverter o estado do contato a ele associado (P1) sempre que houver uma transição em sua entrada (A1), de desenergizado para energizado. Este elemento está associado ao controle de saídas digitais (operadores Oxxxx) e contatos auxiliares (operadores Rxxxx) e só pode ser colocado na oitava coluna da janela de edição.

Figura 4.12 - Representação de um Flip-Flop Oscilador em Ladder

9) Flip-Flop Set de Borda:

O objetivo deste elemento é operar como uma entrada *SET* de um "*Flip-Flop*", que é um elemento básico de memória em circuitos elétricos. Porém, a diferença para este elemento é que ele é ativado somente na borda de subida do sinal de entrada. Este elemento está associado ao controle de saídas digitais (operadores Oxxxx) e contatos auxiliares (operadores Rxxxx) e só pode ser colocado na oitava coluna da janela de edição. Neste tipo de elemento tipo "borda" não há diferença no posicionamento dos elementos de SET e RESET de uma saída digital ou um contato auxiliar no diagrama Ladder devido ao ciclo de varredura do Scan, pois ele é ativado somente na borda de subida do sinal de entrada.

Figura 4.13 - Representação de um Flip-Flop Set de Borda em Ladder

10) Flip-Flop Reset de Borda:

O objetivo deste elemento é operar como uma entrada *RESET* de um "*Flip-Flop*", que é um elemento básico de memória em circuitos elétricos. Porém, a diferença para este elemento é que ele é ativado somente na borda de subida do sinal de entrada. Este elemento está associado ao controle de saídas digitais (operadores Oxxxx) e contatos auxiliares (operadores Rxxxx) e só pode ser colocado na oitava coluna da janela de edição. Neste tipo de elemento tipo "borda" não há diferença no posicionamento dos elementos de SET e RESET de uma saída digital ou um contato auxiliar no diagrama ladder devido ao ciclo de varredura do Scan, pois ele é ativado somente na borda de subida do sinal de entrada.

Figura 4.14 - Representação de um Flip-Flop Reset de Borda em Ladder

11) Contador UP:

O objetivo deste elemento é operar como um contador de eventos. Possui uma entrada para *RESET* do valor de contagem e um parâmetro de limite de contagem. A cada pulso (evento) na entrada de contagem, o valor do contador será incrementado de uma unidade até o limite definido.

Este elemento é representado por um bloco funcional com duas entradas (A1 e A2) e uma saída (B1) e utiliza dois operadores (P1 e P2). O operador P1 que aparece na parte superior do bloco guarda o valor corrente da contagem e deve ser obrigatoriamente uma memória (M). O parâmetro P2 contêm o valor final da contagem, e pode ser uma memória (M) ou uma constante (K).

A entrada inferior (A2) habilita a contagem ("Habilita") e uma transição do estado desenergizado para o estado energizado nesta entrada faz com que o valor corrente da contagem passe para zero (reset do contador). A entrada superior (A1) representa a entrada de contagem ("Conta"). Para contar é necessário que a entrada "Habilita" (A2) esteja energizada e toda vez que a entrada "Conta" (A1) transitar de desenergizada para energizada, o valor corrente da contagem será incrementado de um. A saída (B1) indica se a contagem chegou ao final (contagem = valor final da contagem) ou se o contador está paralisado ou desabilitado (contagem diferente do valor final da contagem).

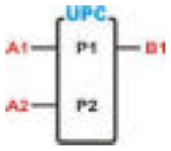

Figura 4.15 - Representação de um Contador UP em Ladder

12) Contador UP-DOWN:

O objetivo deste elemento é contar um determinado número de transições ocorridas na entrada "Conta". Ele conta o número de transições da entrada "Conta" até um certo limite de contagem fornecido pelo usuário como parâmetro. A contagem pode ser crescente (Up) ou decrescente (Down), dependendo do sentido fornecido pelo usuário como entrada para o bloco. A sua saída indica o fim da contagem. Este elemento é representado por um bloco funcional com três entradas e uma saída e precisa de dois operadores. O operador que aparece na parte superior do bloco (P1) guarda o valor corrente do contador e deve ser obrigatoriamente uma memória (operador M) e o inferior (P2) que representa o valor final da contagem, pode ser uma memória ou uma constante.

A entrada inferior (A3) habilita a contagem ("Habilita") e quando esta transitar do estado desenergizado para o energizado, o valor corrente passará a ser zero, se o sentido for 1 (energizado) ou passará a ser igual ao valor inicial da contagem, caso contrário. A entrada do meio do bloco (A2) indica o sentido da contagem, ou seja, se esta entrada for igual a 1 o valor corrente da contagem é incrementado a cada transição da entrada "Conta" e será decrementado caso seja igual a 0. A entrada superior (A1) é a de contagem ("Conta").

Para contar é necessário que a entrada "Habilita" esteja energizada. Cada vez que a entrada "Conta" transitar de desenergizada para energizada o valor de contagem será decrementado ou incrementado de um, dependendo da entrada de sentido. A saída (B1) indica se a contagem chegou ao final (contagem = 0, se sentido = 0 e contagem =valor final, se sentido = 1) ou se o contador está parado ou desabilitado.

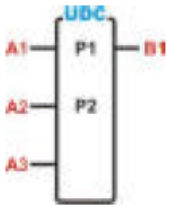

Figura 4.16 - Representação de um Contador UP-DOWN em Ladder

4.3.2- BLOCOS DE MOVIMENTO

1) Movimentação de Variáveis

Este elemento possibilita a inicialização de valores de memórias, a transferência de dados entre memórias, textos, entradas e saídas do PLC e a conversão de tipos de dados entre vários formatos. Este único bloco implementa funcionalidade normalmente encontrada em vários blocos de outros CLP's do mercado. Essencialmente o bloco de movimentação transfere o valor do operando P1 para o operando P2.

Este bloco é necessário para utilizar a entradas/saídas analógica na programação Ladder. Dentre as funções já citadas este bloco ainda pode:

- ✓ Mover o valor da entrada analógica especificada para uma memória inteira ou real.
- ✓ Mover o valor de uma memória inteira para a saída analógica especificada.

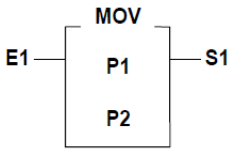

Figura 4.17 - Representação de um bloco de Movimentação de Variáveis em Ladder

Para a utilização deste bloco para entradas analógicas, teremos que:

- **P1:** representa o endereço da entrada analógica utilizada (E0, E1,...,E7).
- **P2:** representa o endereço da memória que armazenará o valor da entrada analógica em função da medida realizada no processo. Este parâmetro pode ser do tipo M ou D obtendo valores de 0 a 4092 (ZAP 500 / ZAP 900).
- **E1:** entrada de habilitação da leitura. Após sua habilitação o bloco passa a movimentar o valor lido na entrada analógica para a memória definida em P2.
- **S1:** saída de operação do canal. Ativa quando o bloco realiza a movimentação do sinal para a memória.

Para a utilização deste bloco para saídas analógicas, teremos que:

- **P1:** representa o endereço da memória contendo o valor a ser escrito na saída analógica. Este valor deve estar entre 0 a 4092. Este valor deve ser proveniente de uma memória inteira (M).
- **P2:** representa o canal de saída analógica utilizado (S0 para ZAP 900).

- **E1:** entrada de habilitação de escrita. Após sua habilitação o bloco passa a movimentar o valor de escrita da memória inteira (M) para a saída analógica definida em P2.
- **S1:** saída de operação do canal. Ativa quando o bloco realiza a movimentação do sinal para a saída analógica.

2) Movimentação Indexada

Possibilita a movimentação de um grupo de memórias inteiras onde a fonte e o destino são indexados pelos parâmetros P1 e P2 consecutivamente. O valor numérico presente na memória indicada pelo parâmetro P1 será a memória fonte. O valor numérico presente na memória indicada pelo parâmetro P2 será a memória destino. O valor numérico presente na memória indicada pelo parâmetro P3 especifica a quantidade de dados que serão transferidos a partir da memória fonte para a memória de destino.

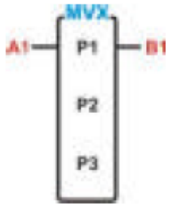

Figura 4.18 - Representação de um bloco de Movimentação Indexada em Ladder

Por exemplo, na figura abaixo, quando A1 é ativado, serão transferidos 2 valores de memórias consecutivas da fonte (M06) para o destino (M10). Após isso a saída B1 é energizada.

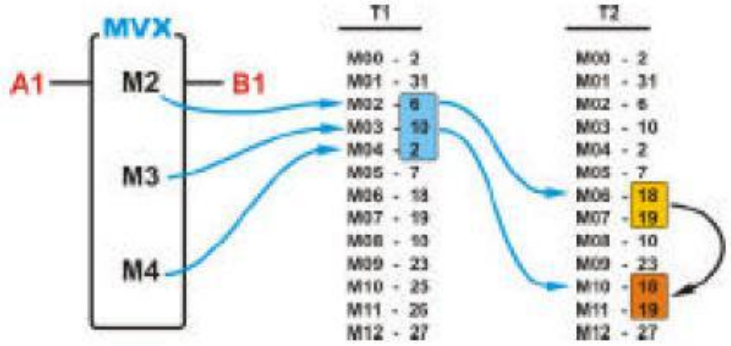

Figura 4.19 - Exemplo utilizando Movimentação Indexada

3) Inicialização de Dados

Possibilita inicializar um bloco de memórias ou contatos com um valor predefinido. O parâmetro P1 especifica uma memória cujo conteúdo identifica a memória inicial do bloco a ser inicializado. O parâmetro P2 especifica qual tipo de variável e o valor a ser utilizado na inicialização do bloco. O conteúdo da memória especificado em P3, define o tamanho do bloco, ou seja, a quantidade de dados que serão inicializados a partir da variável especificada em P1. O bloco é ativado quando A1 é energizado. Quando a saída B1 for energizada, o processo de inicialização do bloco de memórias ou contatos foi realizado com sucesso.

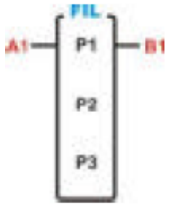

Figura 4.20 - Representação de um bloco de Inicialização de Dados em Ladder

4) Troca de Dados

Efetua troca de valores entre dois operadores especificados. P1 especifica a variável 1 a ter seu conteúdo trocado com o conteúdo da variável 2. P2 especifica a variável 2 a ter seu conteúdo trocado com o conteúdo da variável 1.

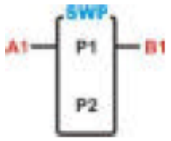

Figura 4.21 - Representação de um bloco de Troca de Dados em Ladder

Os parâmetros P1 e P2 podem ser do tipo M ou D. Quando os parâmetros são do mesmo tipo, o resultado da operação transfere o valor atual de P1 para P2 e vice versa. Quando P1 é inteiro (M) e P2 é real (D), os valores de P1 e P1+1 são agrupados e transferidos para P2. O valor de P2 é dividido e transferido para P1 e P1+1. Quando P1 e real e P2 é inteiro, operação equivalente é efetuada. Por exemplo, considere o bloco indicado a seguir, onde o conteúdo de M1 será trocado com o conteúdo de D5. Nesta situação como as memórias D utilizam 32 bits para sua representação e a memórias M utilizam apenas 16 bits, são necessárias 2 memórias M para receberem o conteúdo da memória D. As tabelas a seguir indicam o conteúdo de cada memória em representação hexadecimal antes da execução do bloco e depois da mesma.

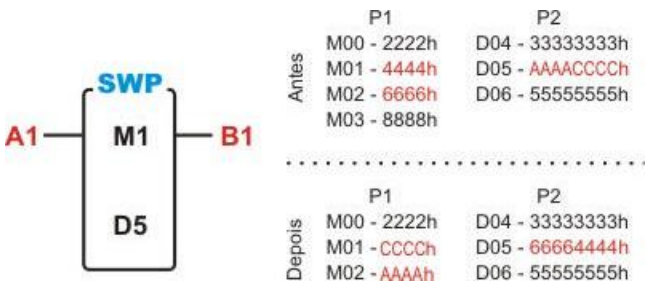

Figura 4.22 - Exemplo utilizando Troca de Dados

4.3.3- BLOCOS DE MATEMÁTICA

1) Somador, Subtrador, Multiplicador, Divisor

Este elemento efetua a operação matemática (de sua funcionalidade) entre os operandos P1 e P2, armazenando o resultado em P3. Opera com valores inteiros ou reais, efetuando automaticamente a conversão de tipos quando necessário. Quando P1 e P2 são de tipos diferentes, antes de efetuar a operação, o elemento converte o operando P2 de forma a efetuar a operação sempre com operandos do mesmo tipo (inteiro ou real). Assim, o tipo de operação a ser realizada (inteira ou real) será sempre definida pelo tipo do operando P1.

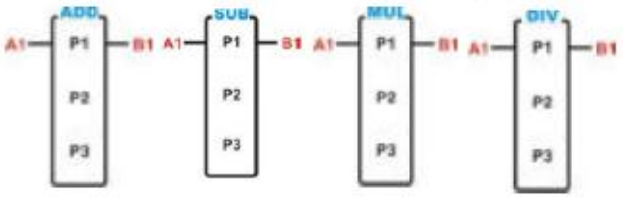

Figura 3 - Representação dos blocos: Somador, Subtrador, Multiplicador e Divisor em Ladder

2) Extrator de Raiz Quadrada, Log na Base 10, Exponenciação

Este elemento efetua a operação matemática (de sua funcionalidade) do parâmetro P1, transferindo o resultado para P2. Opera apenas com valores reais.

Figura 4 - Representação dos blocos: Extrator de raiz quadrada, Exponenciação e Log na base 10 em Ladder

3) Potenciação

Este elemento efetua a operação matemática de potenciação entre os parâmetros P1 e P2, transferindo o resultado para o parâmetro P3. Opera apenas com valores reais.

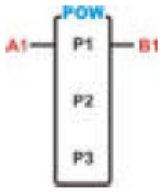

Figura 5 - Representação de um bloco de Potenciação em Ladder

4) AND, OR, XOR

Este elemento efetua a operação lógica (de sua funcionalidade) entre os parâmetros P1, P2, transferindo o resultado para P3. Opera apenas com valores Inteiros. Esta operação é efetuada bit a bit entre os conteúdos de P1 e P2, ou seja, os Bits 0 de P1 e P2 são operados e o resultado transferido para o bit 0 de P3. O mesmo procedimento se repete para todos os 16 bits dos operandos inteiros.

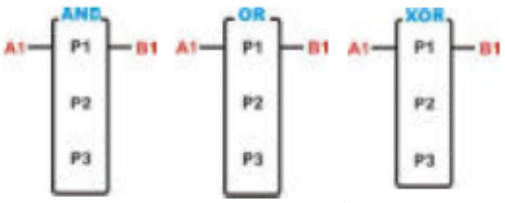

Figura 6 - Representação dos blocos: AND, OR e XOR em Ladder

5) Deslocamento para Direita, Deslocamento para Esquerda

Este elemento efetua a operação lógica Deslocamento para direção direita/esquerda entre os parâmetros P1 e P2, transferindo o resultado para P3. Opera apenas com valores Inteiros. Esta operação é efetuada deslocando para direita/esquerda bit a bit o conteúdo de P1 pelo número de vezes definido no operando P2.

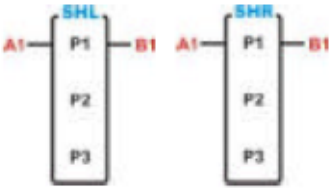

Figura 7 - Representação dos bloco de Deslocamento para: esquerda e direita em Ladder

4.3.4- BLOCOS DE COMPARAÇÃO

1) Teste se: P1 = P2, P1 ≠ P2, P1 > P2, P1 ≥ P2, P1 < P2, P1 ≤ P2

O objetivo destes elementos é realizar a comparação entre operadores. Esta comparação é do tipo que compara P1 e P2.

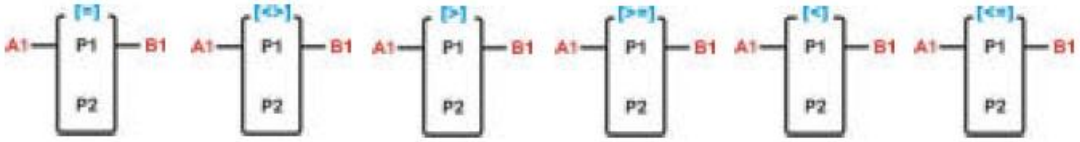

Figura 8 - Representação dos blocos de Teste: P1 = P2, P1 ≠ P2, P1 > P2, P1 ≥ P2, P1 < P2, P1 ≤ P2 em Ladder

2) Teste Lógico

O objetivo deste elemento é realizar a operação lógica *AND* (E) *bit* a *bit* entre dois operadores. Este elemento é representado por um bloco funcional com uma entrada, uma saída e tem dois operadores. O operador P1 deve ser uma memória inteira (operador M) ou uma constante inteira (operador K). O segundo parâmetro pode ser uma memória inteira (operador M) ou uma constante inteira (operador K), porem não devemos configurar os parâmetros P1 e P2 com constantes inteira (operador K), pois o próprio nome do operando já diz:"constante", não podendo efetuar a lógica *AND* bit a bit entre si. A entrada do bloco (A1) habilita a operação ("Habilita"). Para que a operação se realize é necessário que a entrada "Habilita" esteja energizada.

É realizada uma operação lógica *AND bit* a *bit* entre os operadores P1 e P2 e o resultado é refletido na saída B1. A saída B1 indica o resultado da operação. Se energizada, o resultado do teste lógico foi diferente de zero, desenergizada caso contrário.

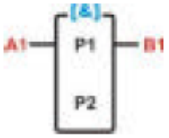

Figura 9 - Representação de um bloco de Teste lógico em Ladder

4.3.5- BLOCOS DE FLUXO

1) Início de Relé Mestre, Fim de Relé Mestre

O objetivo destes elementos é realizar a comparação entre operadores. Esta comparação é do tipo que compara P1 e P2.

O objetivo destes elementos é delimitar uma área do programa que será executada sempre que o bloco **Relé Mestre** estiver ativado.

Basicamente o bloco **Relé Mestre** tem dupla utilidade, pois marca o início da área a ser executada e ao mesmo tempo permite que esta mesma área do programa seja executada quando o bloco estiver energizado. Este bloco utiliza o bloco Fim de Relé Mestre como delimitador final da área de programa que deverá ser executada. Um programa pode ter quantos relés mestre se desejar.

- **Relé Mestre** - Este elemento é representado por um bloco de uma entrada. Este elemento funciona como uma bobina, ou seja, quando tiver energizada a sua entrada (**A1**) a área de programa delimitada por ele e o bloco **Fim de Relé Mestre** será executada, caso contrário esta área do programa não será executada. Este elemento deve obrigatoriamente ocupar a oitava coluna da janela de edição. Com a entrada A1 ativa, toda área de programação do Inicio do Relé Mestre ate o Fim de Relé Mestre será executada. B1 ativo indica que o Inicio de Relé Mestre foi energizado.
- **Fim de Relé Mestre** - Este elemento é representado por um bloco sem entrada. Este elemento funciona como um delimitador de área de programa, ou seja, o ponto do programa em que se encontra delimita a área de atuação do **Relé Mestre** imediatamente anterior a ele, não possuindo qualquer outra funcionalidade. Este elemento deve obrigatoriamente ocupar a oitava coluna da janela de edição. Com a entrada A1 ativa, será finalizada a execução do Relé Mestre. B1 ativo, indica que Fim de Relé Mestre foi energizado.

Se na área delimitada pelo Relé Mestre e Fim de Relé Mestre existir alguma bobina e o Relé Mestre estiver desenergizada, então as bobinas estarão resetadas (desligadas),independentemente do estado das lógicas associadas a elas.

Figura 10 - Representação dos blocos de: Início de relé mestre e fim de relé mestre em Ladder

2) Início de Bloco de Lógica, Fim de Bloco de Lógica

O objetivo destes elementos é delimitar o início e o fim de uma área do programa que será executada sempre que o Bloco de Lógica estiver ativado. Estes blocos sempre devem ser utilizados em conjunto. O bloco Início de Bloco de Lógica delimita o início e o bloco Fim de Bloco de Lógica delimita o fim da área de programa que deverá ser executada quando o Bloco de Lógica correspondente estiver energizado. Um programa pode ter quantos Blocos de Lógica se desejar. Fazendo uma analogia com os termos de programação, o Bloco de Lógica seria uma chamada de uma rotina, ao passo que o bloco Início de Lógica corresponderia ao delimitador de início da rotina e o bloco de Fim de Lógica o seu delimitador de fim de rotina. Para inseri-lo no programa deve-se posicionar o cursor na posição desejada, selecionar no menu à esquerda o grupo "Fluxo" e clicar no botão correspondente.

No bloco Início de Bloco de Lógica**:**

- A entrada A1 no estado ativa implica na execução do Bloco de Lógica. A entrada A1 somente será energizada quando existir uma chamada (através de um bloco BLQ) para este respectivo bloco de lógica durante a execução da aplicação Ladder.
- B1 ativado implica que o Inicio do Bloco de Lógica foi energizado.
- Txxxx - Identifica o Bloco de Lógica que este bloco está delimitando o início. Este valor tem que ser um valor maior ou igual a zero. Quando este valor é zero, o Bloco de Lógica será executado automaticamente toda vez em que o controlador for ligado, servindo como uma rotina de Inicialização do programa. Quando este valor for diferente de zero, o bloco de Lógica será executado apenas quando o Bloco de Lógica correspondente estiver energizado.
- Este elemento é representado por um bloco com uma entrada, uma saída e um parâmetro. Ele funciona como habilitação de execução e um Bloco de Lógica delimitado pelos blocos Início de Bloco de Lógica e Fim de Bloco de Lógica. Quando energizado, o Bloco de Lógica identificado pelo parâmetro Txxxx será executado, ou não, caso contrário

No bloco Fim de Bloco de Lógica**:**

- O bloco de lógica retorna à energização resultante do processamento da lógica Ladder especificada na linha em que se encontra este delimitador de fim de o Bloco de Lógica. Se não for especificada nenhuma lógica nesta linha retorna-se energização “ativa”, caso contrário, retorna a energização resultante do processamento da lógica Ladder (ativa ou inativa).

- A saída do Bloco de Lógica corresponderá à energização A1 resultante da linha deste delimitador de fim de bloco de lógica.
- **Txxxx** - Identifica o **Bloco de Lógica** que este bloco está delimitando o seu respectivo fim. Este valor tem que ser um valor maior ou igual a zero. Quando este valor é zero, o Bloco de Lógica será executado automaticamente toda vez em que o controlador for ligado, servindo como uma rotina de Inicialização do programa. Quando este valor for diferente de zero, o bloco de Lógica será executado apenas quando o Bloco de Lógica correspondente estiver energizado.
- Este elemento é representado por um bloco sem entrada. Este elemento funciona como um delimitador de área de programa, ou seja, o ponto do programa em que se encontra delimita o fim da área de atuação do Bloco de Lógica que ele representa, não possuindo qualquer outra funcionalidade. Este elemento deve obrigatoriamente ocupar a oitava coluna da janela de edição e deve ser posicionado sempre após o delimitador de fim de programa (bloco de Fim de Programa) e também após o delimitador de Início de Bloco de Lógica correspondente.

Figura 11 - Representação dos blocos de: Início de bloco de lógica e fim de bloco de lógica em Ladder

3) Bloco de Lógica

Este elemento executa o bloco de lógico desenvolvido depois do elemento Fim de Programa.

- Sempre que a entrada A1 estiver energizada o Bloco de Lógica identificado pelo parâmetro Txxxx será executado, ou não, em caso contrário.
- A saída B1 segue o valor da energização da linha em que se encontra o bloco Fim de Bloco de Lógica. Se esta linha estiver energizada, B1 também estará, e vice e versa.
- Txxxx - Identifica o Bloco de Lógica a ser executado pelo programa. Este valor tem que ser um valor maior ou igual a zero. Quando este valor é zero, o Bloco de Lógica será executado automaticamente toda vez em que o controlador for ligado, servindo como uma rotina de Inicialização do programa. Quando este valor for diferente de zero, o bloco de Lógica será executado apenas quando o Bloco de Lógica correspondente estiver energizado.

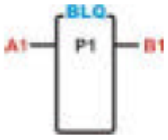

Figura 12 - Representação de um Bloco de lógica em Ladder

4) Fim de Programa

O objetivo deste elemento é delimitar o final da área do programa principal da aplicação Ladder. Este bloco sempre deve estar presente em um programa, sinalizando o fim do mesmo. A entrada A1 energizada indica o fim de execução do programa principal.

Figura 4.33 - Representação de um bloco de Fim de programa em Ladder

4.4- EXEMPLOS DE UTILIZAÇÃO DO LADDER

4.4.1- COMANDO DE UMA LÂMPADA POR UMA CHAVE ÚNICA

Neste exemplo, o Ladder desenvolvido representa o comando de uma lâmpada por uma chave liga/desliga semelhante ao exemplo da seção 3.5.1.

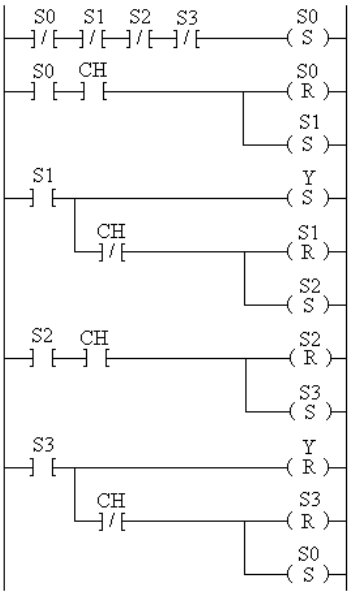

Figura 4.34 - Ladder referente ao comando de uma lâmpada por uma chave única

4.4.2- LIGAR UMA LÂMPADA DURANTE UM TEMPO

Neste exemplo, o Ladder desenvolvido representa uma lâmpada que ao ser ligada, permanece assim por 5s antes de ser desligada novamente. Este exemplo é semelhante ao exemplo da seção 3.5.2.

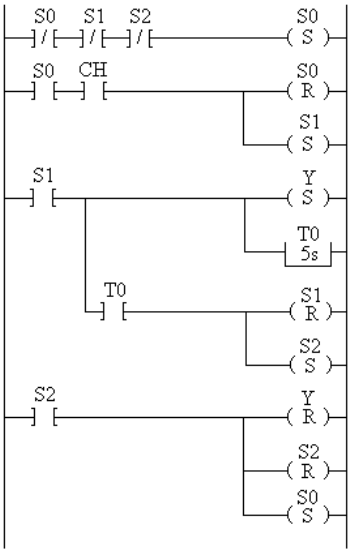

Figura 4.35 – Ladder referente ao comando de ligar uma lâmpada durante um certo tempo

4.4.3- PARTIDA SEQUENCIAL DE MOTORES

Uma aplicação interessante do temporizador consiste na partida sequencial de motores. Para exemplificar pode-se fazer um programa para acionar 3 motores de forma que o primeiro seja acionado imediatamente, o segundo deve ser acionado 5s após o primeiro e o terceiro 8s após o primeiro. O programa é apresentado abaixo, nota-se que a aplicação é simples e direta. Esta estratégia é muito utilizada para evitar picos de consumo de energia e corrente, caso os motores partissem todos simultaneamente.

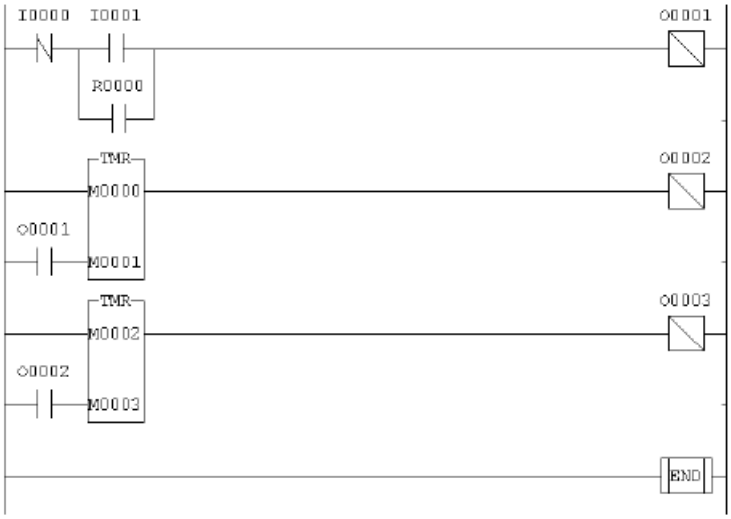

Figura 4.36 - Ladder referente à partida sequencial de motores

4.4.4- SEGURANÇA NO COMANDO DE PRENSAS

No comando de prensas antigamente utilizavam-se duas botoeiras em série para evitar que o operador estivesse com uma mão livre ao acionar a mesma. Entretanto eles começaram a prender um dos botões com um peso, tornando a estratégia ineficaz. Nos comandos modernos faz-se com que as botoeiras devam ser acionadas em um intervalo de tempo menor que 3s, por exemplo, caso contrário a prensa não aciona. O programa em Ladder para acionamento é mostrado na figura abaixo:

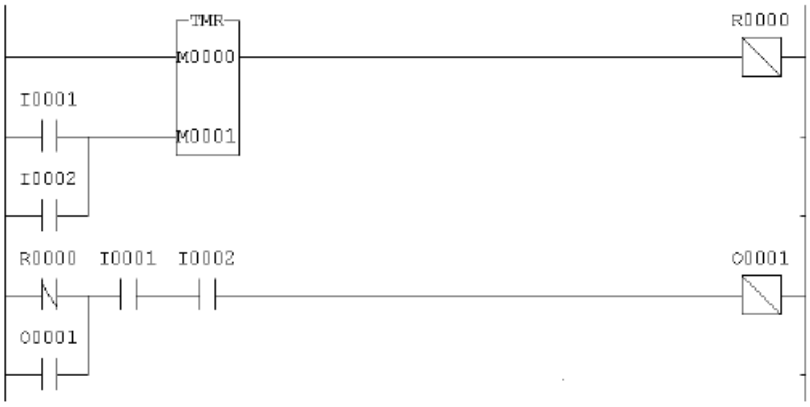

Figura 4.37 - Ladder referente ao sistema de segurança no comando de prensas

4.4.5- SISTEMA DE PARTIDA DIRETA DE UM MOTOR

Como já foi discutido anteriormente, linguagem Ladder é o sistema de representação que mais se assemelha à tradicional notação de diagramas elétricos. Dessa forma, fica muito fácil passar um diagrama elétrico para linguagem Ladder. Basta transformar as colunas em linhas, como se mostra nas duas próximas figuras, para o caso de uma simples partida direta.

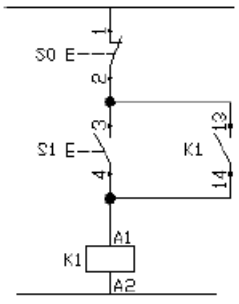

Figura 4.38 - Diagrama elétrico do sistema de partida direta de um motor

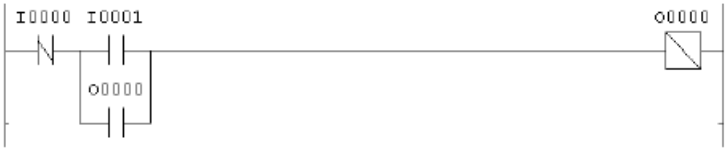

Figura 4.39 - Ladder referente ao sistema de partida direta de um motor

4.4.6- SISTEMA DE PARTIDA COM REVERSÃO DE UM MOTOR

Da mesma forma do exemplo anterior, basta transformar as colunas em linhas, como se mostra nas duas próximas figuras, para o caso de uma partida com reversão.

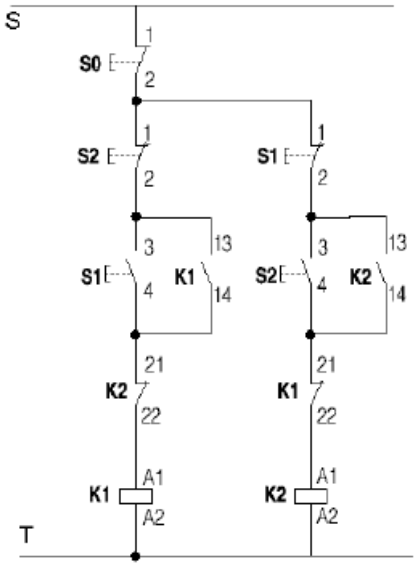

Figura 4.40 - Diagrama elétrico do sistema de partida com reversão de um motor

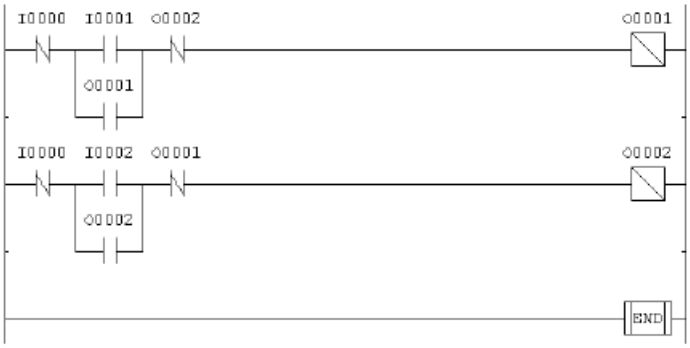

Figura 4.41 - Ladder do sistema de partida com reversão de um motor

5- SUPERVISÓRIO INTOUCH

5.1- INTRODUÇÃO AO SISTEMA SUPERVISÓRIO

O supervisório é um software destinado a promover a interface homem/máquina, proporcionando a supervisão de um processo através de telas devidamente configuradas. Possui telas que representam o processo que podem ser animadas em função das informações recebidas pelo CLP, controlador, etc. Por exemplo: no acionamento de uma bomba, a representação na tela mudará de cor informando que a mesma está ligada; se um determinado nível varia no campo, a representação na tela mudará informando a alteração de nível. O supervisório lê e escreve na memória do CLP ou controlador para a atualização das telas.

O elo de comunicação entre a IHM e o PLC (ou outro equipamento de controle monitorado) normalmente se dá por meio de um protocolo de comunicação específico que reproduz na IHM as variáveis do processo através de Tags – o protocolo OPC. Assim, uma Tag representa, em última análise, uma variável na IHM que pode ser do tipo discreto, numérico ou alfanumérico. Devido à bidirecionalidade do sentido de comunicação entre PLC e IHM, uma Tag pode tanto monitorar o status do controlador, como também enviar valores (ou setpoints – valores predefinidos) a ele.

As IHMs podem liberar a CPU do controlador da monitoração de situações anômalas do processo pela geração de sinais de alarme. Tal procedimento é feito pela constante monitoração das Tags suscetíveis a falhas por parte da IHM que podem, inclusive, ter a vantagem de, em ocorrências, sugerir ao operador que providências devem ser tomadas diante do defeito ocorrido.

Em situação similar, a IHM pode simplesmente alertar o operador sobre a troca de situação do processo, por meio da geração de sinais de eventos que são obtidos pela monitoração das variáveis pertinentes.

Esses recursos por si mesmo já justificariam o uso de uma IHM. Porém, mais do que isto, em grande parte dos casos, permitem ainda o registro histórico (pelo armazenamento de dados) dos eventos e alarmes, de forma a permitirem posterior a análise de ocorrências pela equipe e engenharia de processo, permitindo a obtenção de dados para controle estatístico, bem como de análise históricos para consulta, plotagem e diversos outros tipos de relatório e gráficos de tendência.

É exatamente por intermédio de arquivos de receitas, histórica, linguagem de programação interativa (scripts), entre outros, que os softwares de supervisão e controle devem se comunicar com diferentes tipos de bancos de dados, tal que lhe possibilitem enviar ou receber informações geradas para o chão de fábrica. Isto é possível por meio da ferramenta ODBC (Open Data Base Connectivity) que permite enviar informações vindas da rede corporativa automaticamente para o processo.

Outras tecnologias estão facilitando e aumentando a eficiência, o grau de flexibilidade e de controle do processo fabril, pelos sistemas de supervisão, como, por exemplo, a tendência da engenharia de software de desenvolvimento de programas componentizados (módulos que se integram perfeitamente com todo o sistema); recursos de captura, registro e transmissão digital de imagens em tempo real; sistemas de supervisão, gerenciamento e distribuição de informações por meio da Internet; entre outros.

Um recurso interessante disponível nas IHMs é o armazenamento de conjuntos de setpoints, gravados em arquivos que, ao serem enviados ao controlador, definem as diversas parametrizações do sistema, ou receitas de produção. Assim, torna-se possível, por exemplo, alterar rapidamente a produção de um determinado tipo de peça para um outro tipo, ao simples enviar de um único comando por parte do operador.

Atualmente o mercado disponibiliza vários programas supervisórios, tais como INTOUCH, ELIPSE, AIMEX, FIX-32, VIEW, CIMPLIST e outros. Apresentaremos neste capítulo o Intouch, um dos mais populares supervisórios na indústria.

O Intouch é um conjunto de softwares que se destina à criação de telas gráficas de interação com CLP's, controladores Multiloop, Fieldbus, etc. É uma das interfaces homem-máquina mais simples de ser configurada em comparação aos demais softwares existentes no mercado. Abaixo temos um diagrama básico de como um sistema supervisório está ligado a um CLP.

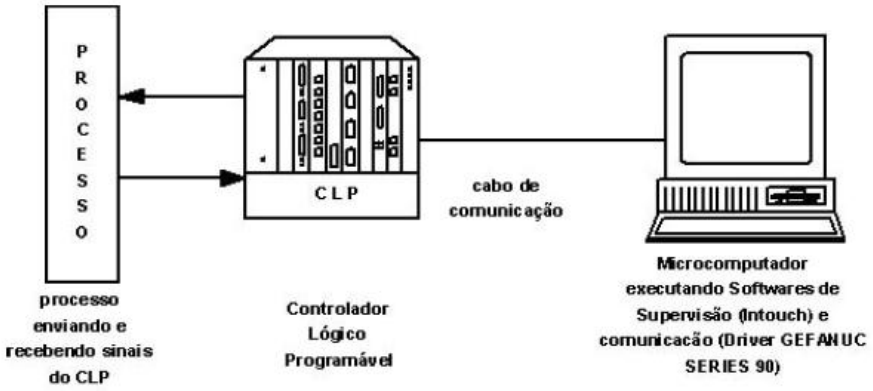

Figura 5.1 - Exemplo de diagrama básico de interligação com um CLP

O Intouch é caracterizado por dois modos:

- Desenvolvimento – "Windows Maker": responsável pela criação das aplicações e janelas, e edição das animações
- Execução – "Windows Viewer": executa a produção do Windows Maker.

5.2- GUIA SOBRE O INTOUCH

5.2.1- CRIANDO UMA APLICAÇÃO

1) Depois de instalado o software do *Intouch*.
2) No menu iniciar, programa, *Wonderware FactorySuite*.

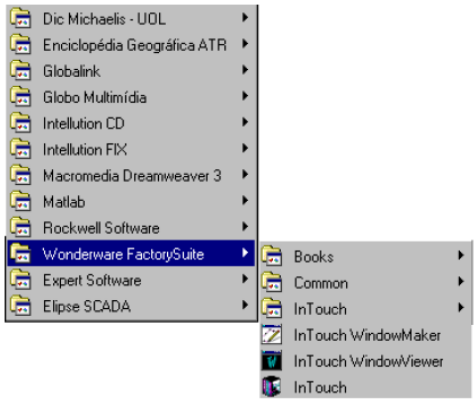

Figura 5.2 - Encontrando o Intouch no Windows

3) *Double-click* no icone.
4) Aparecerá o gerenciador de aplicativos, e através deste é possível cria uma nova aplicação, alterar a lista de aplicativos, etc.

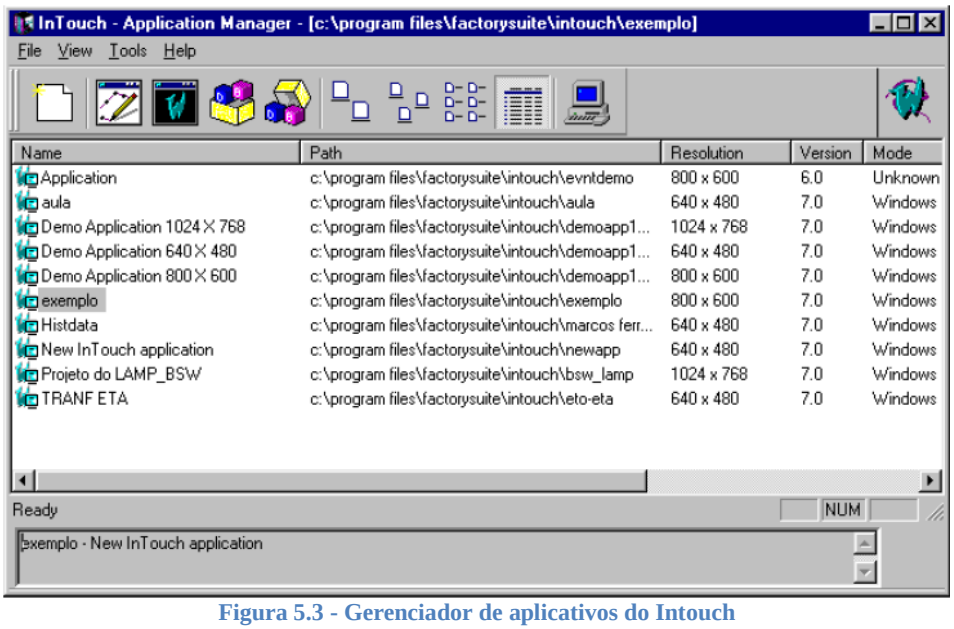
InTouch - Application Manager - [c:\program files\factorysuite\intouch\exemplo]
File View Tools Help
Name Path Resolution Version Mode
Application c:\program files\factorysuite\intouch\evntdemo 800 x 600 6.0 Unknown
aula c:\program files\factorysuite\intouch\aula 640 x 480 7.0 Windows
Demo Application 1024 X 768 c:\program files\factorysuite\intouch\demoapp1... 1024 x 768 7.0 Windows
Demo Application 640 X 480 c:\program files\factorysuite\intouch\demoapp1... 640 x 480 7.0 Windows
Demo Application 800 X 600 c:\program files\factorysuite\intouch\demoapp1... 800 x 600 7.0 Windows
exemplo c:\program files\factorysuite\intouch\exemplo 800 x 600 7.0 Windows
Histdata c:\program files\factorysuite\intouch\marcos ferr... 640 x 480 7.0 Windows
New InTouch application c:\program files\factorysuite\intouch\newapp 640 x 480 7.0 Windows
Projeto do LAMP_BSW c:\program files\factorysuite\intouch\bsw_lamp 1024 x 768 7.0 Windows
TRANF ETA c:\program files\factorysuite\intouch\eto-eta 640 x 480 7.0 Windows
Ready
NUM
exemplo - New InTouch application

Figura 5.3 - Gerenciador de aplicativos do Intouch

5) Selecione o botão *NEW*, ou clique em *File/New*, em seguida clique em avançar.
6) Entre com o nome do diretório onde sua aplicação será armazenada, em seguida clique em avançar.
7) Defina nome da aplicação e descrição, em seguida clique em concluir.

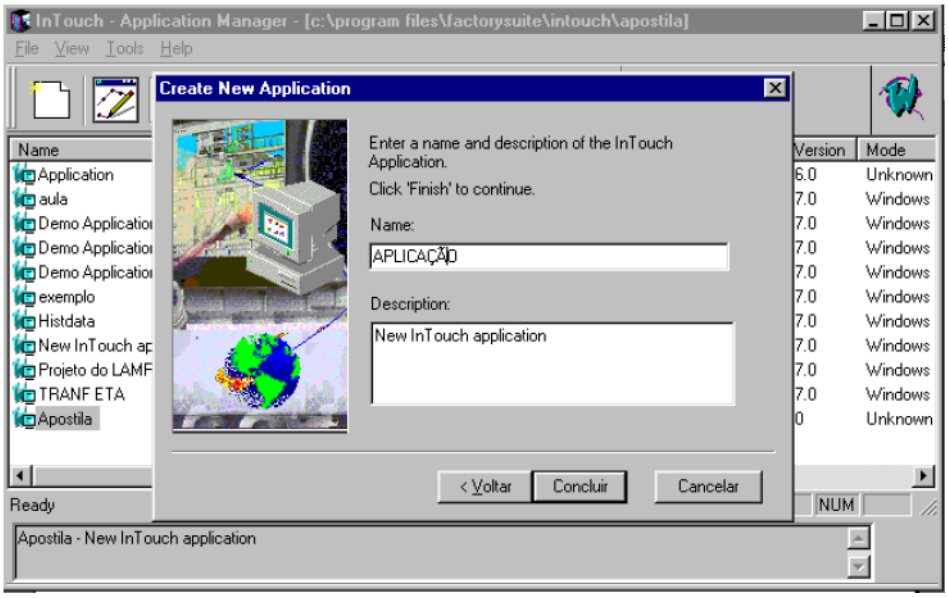

Figura 5.4 - Tela de criação de nova aplicação no Intouch

Observações:

- Cada aplicação deverá ser armazenada em um diretório distinto que se encontrará no diretório do *INTOUCH*.
- O nome da aplicação fica a gosto do cliente
- Os arquivos do diretório da aplicação não podem ser apagados (a não ser pelo próprio *InTouch*) , exceto os arquivos *.?bk.
- Ao fazer backup, todo o diretório deve ser copiado.
- Ao restaurar, deve-se fazê-lo em um diretório vazio.

5.2.2- INCLUINDO UMA APLICAÇÃO JÁ EXISTENTE

Caso já tenha uma aplicação pronta para inclui-la em um outro computador basta copiar o diretório para o local que deseja e clicar no *Intouch*, selecionando no menu *TOOLS*, em seguida *FIND APPLICATIONS*. Aparecerá uma "dialog box", permitindo adicionar um novo diretório de aplicação.

Quando uma aplicação é excluída da lista, não é deletada do HD, é necessário que o faça via explorer. O mesmo acontece na inclusão de uma nova aplicação. Os diretórios que contém aplicações do *Intouch* precisam estar nesta lista para serem acessados.

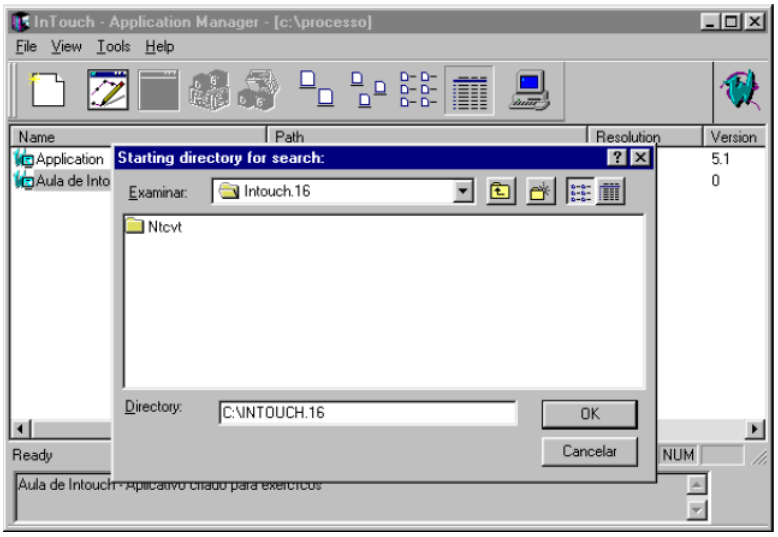

Figura 5.5 - Tela de inclusão de aplicação já existente

5.2.3- OUTRAS INFORMAÇÕES SOBRE A JANELA INTOUCH

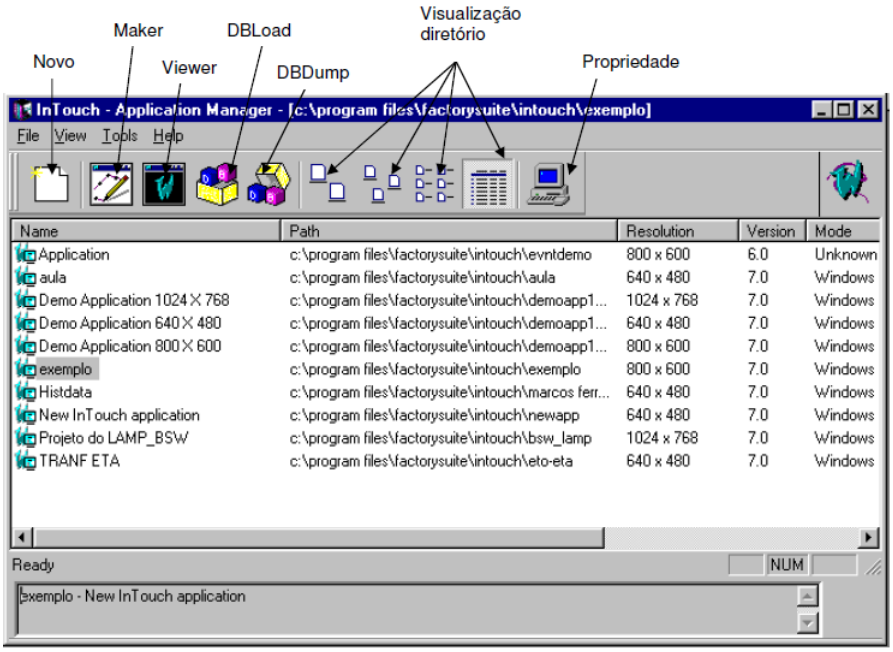
Novo
Maker
Viewer
DBLoad
DBDump
Visualização diretório
Propriedade
InTouch - Application Manager - [c:\program files\factorysuite\intouch\exemplo]
File View Tools Help
Name
Path
Resolution
Version
Mode
Application c:\program files\factorysuite\intouch\evntdemo 800 x 600 6.0 Unknown
aula c:\program files\factorysuite\intouch\aula 640 x 480 7.0 Windows
Demo Application 1024 X 768 c:\program files\factorysuite\intouch\demoapp1... 1024 x 768 7.0 Windows
Demo Application 640 X 480 c:\program files\factorysuite\intouch\demoapp1... 640 x 480 7.0 Windows
Demo Application 800 X 600 c:\program files\factorysuite\intouch\demoapp1... 800 x 600 7.0 Windows
exemplo c:\program files\factorysuite\intouch\exemplo 800 x 600 7.0 Windows
Histdata c:\program files\factorysuite\intouch\marcos ferr... 640 x 480 7.0 Windows
New InTouch application c:\program files\factorysuite\intouch\newapp 640 x 480 7.0 Windows
Projeto do LAMP_BSW c:\program files\factorysuite\intouch\bsw_lamp 1024 x 768 7.0 Windows
TRANF ETA c:\program files\factorysuite\intouch\eto-eta 640 x 480 7.0 Windows
Ready
NUM
exemplo - New InTouch application

Figura 5.6 - Outras informações sobre a janela Intouch

- ***Novo*** – cria uma nova aplicação.
- ***Maker*** – destina-se a criação de janelas e edição das animações.
- ***Viewer*** – é o software que executa a janela que foi "produzida" no Windows Maker.
- ***DBLoad*** – exporta a base de dados na forma .CSV
- ***DBDump*** – importa a base de dados.
- ***Visual ização dos Diretórios*** – Podemos ver os diretórios como ícone pequeno, grande, em lista ou com detalhes.
- ***Propriedades*** – são as propriedades do Intouch.

5.3- GUIA SOBRE O WINDOWS MAKER

5.3.1- CRIANDO JANELAS

Marque o diretório da aplicação que deseja trabalhar, em seguida selecione o ícone do *Window Maker* no gerenciador de aplicativos do Intouch.

Obs: caso não possua chave de hardware, selecione a opção para ignorar.

Crie a primeira janela da aplicação, chamando o comando /*File/New Window* ou clicando no botão *New Window*.

Surgirá uma "dialog box", para definição das propriedades da janela a ser criada.

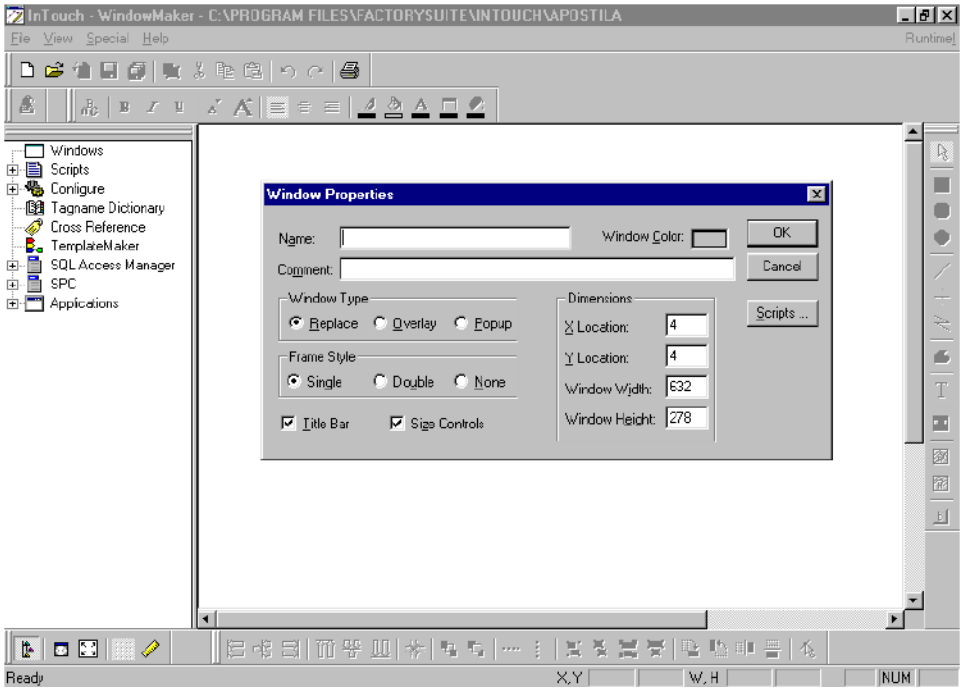

Figura 5.7 - Tela inicial do Windows Maker

5.3.2- PROPRIEDADES DE UMA JANELA

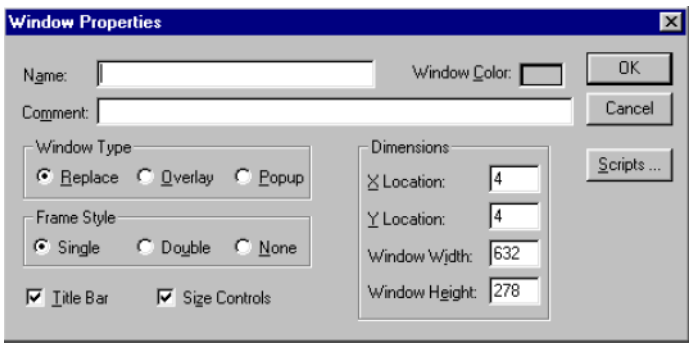

Figura 5.8 - Tela de definições de propriedades de uma janela do Windows Maker

- ***Name:*** nome da janela.
- ***Window Color:*** cor de fundo da janela.
- ***Comment:*** comentário associado à janela (opcional).
- ***Window Type:*** tipo da janela:
    - ***Replace:*** fecha qualquer janela que haja interseção parcial ou total;
    - ***Overlay:*** aparece sobre a janela corrente;
    - ***Popup:*** aparece sempre sobre qualquer janela aberta por cima de tudo (uma de cada vez).
- ***Frame Style:*** tipo de moldura da janela:
    - ***Single:*** moldura simples;
    - ***Double:*** moldura dupla;
    - ***None:*** sem moldura.
- ***Title Bar:*** janela com título.
- ***Size Controls:*** habilita o controle de redimensionamento da janela:
    - ***X Location:*** posição horizontal (em pixels);
    - ***Y Location:*** posição vertical (em pixels);
    - ***Window Width:*** largura da janela (em pixels);
    - ***Window Height:*** altura da janela (em pixels);
- **Scripts:** associa ações a serem executadas em 3 situações:
    - ***On Show*** - quando a janela é aberta;
    - ***While Showing*** - enquanto a janela estiver aberta numa frequência específica;
    - ***On Hide*** - quando a janela é fechada.

5.3.3- DESCRIÇÃO DAS FERRAMENTAS

A seguir serão descritas as ferramentas que compõem o ambiente de trabalho do *WindowMaker*.

Ícone	Descrição
	Seleção de objetos, multiplos ou específicos com a tecla SHIFT acompanhada.
	Retângulo
	Retângulo com bordas arredondadas
	Elipse
	Linha

	Linha vertical ou horizontal
	Seqüência de linhas
	Polígono
	Texto
	Importação de bitmap
	Gráfico de Tendência Real
	Gráfico de Tendencia Histórica
	Botão
	Seleciona e altera tipo de fonte.
	Altera fonte do texto selecionado para formato Bolbo
	Altera fonte do texto selecionado para Itálico
	Altera fonte do texto selecionado para Sublinhado
	Diminui tamanho da fonte do texto selecionado quando a mesma é tipo True Type (TT)
	Aumenta tamanho da fonte do texto selecionado quando a mesma é tipo True Type (TT)
	Alinha texto a esquerda
	Centraliza texto
	Alinha texto a direita
	Altera cor de linha do objeto selecionado
	Altera cor de enchimento do objeto selecionado.
	Altera cor da fonte selecionada
	Altera cor da janela
	Altera trasparencia
	Habilita/Desabilita explorer do Intouch
	Hide/Show todas as ferramentas
	Mostra janela editada em formato full screen
	Habilita/Desabilita grade da janela editada
	Habilita /Desabilita régua do WindowMaker
	Cria nova janela
	Abre janela
	Fecha janela

	Salva janela corrente para o disco
	Salva todas as janela para o disco
	Duplica objeto selecionado
	Recorta
	Copia para área de transferência
	Cola da área de transferência
	Retorna ação
	Avança ação retornada
	Imprime
	Alinha objetos selecionados à esquerda
	Alinha objetos selecionados no meio horizontal
	Alinha objetos selecionados à direita
	Alinhamento superior de objetos selecionados
	Alinhamento no meio vertical de objetos selecionados
	Alinhamento inferior de objetos selecionados
	Alinhamento central de objetos selecionados
	Enviar para trás
	Trazer para frente
	Espaçamento horizontal igual
	Espaçamento vertical igual
	Agrupara em símbolo
	Desagrupar símbolo
	Agrupar célula
	Desagrupar célula
	Girar 90 graus sentido horário
	Gerar 90 graus sentido anti-horário
	Espelhar horizontal
	Espelhar Vertical
	Selecionar Wizard, objetos previamente editados e configurados.

5.3.4- MENU DO WINDOWS MAKER

File

COMANDO	TECLAS DE ATALHO	AÇÃO
New Window...	Ctrl+N	Cria uma nova janela (janela em branco).
Open Window...	Ctrl+O	Abre uma aplicação já existente, através de um dialog-box
Save Window...	Ctrl+S	Salva a aplicação corrente, através de um dialog-box
Close Window...		Fecha a aplicação corrente, através de um dialog-box
Delete Window...		Apaga a(s) tela(s), através de um dialog-box.
Save Window As...		Salva janela aberta com outro nome.
Save All Windows		Salva Todas as Janelas.
WindowView..		Executa(Run) a aplicação corrente.
Print...		Opções para imprimir, através de um dialog-box.
Export Window...		Exportar a janela da aplicação corrente para outra, através de um dialog-box.
Import...		Importa a janela de outra aplicação, através de um dialog-box.
Exit		Sai do Supervisório

Edit

COMANDO	TECLAS DE ATALHO	AÇÃO
Undo Create	Ctrl+Z	Desfaz a última ação.
Nothing to Redo	Ctrl+Y	Refaz a última ação.
Duplicate	Ctrl+D	Duplica o(s) objeto(s) selecionado(s).
Cut	Ctrl+X	Copia o(s) objeto(s) selecionados (s) para o Clipboard e apaga-o(s) janela.
Copy	Ctrl+C	Copia o(s) objeto(s) selecionado (s) para o Clipboard.
Paste	Ctrl+V	Insere o(s) objeto(s) do Clipboard na janela.
Erase	Del	Elimina o(s) objeto(s) selecionado(s).
Impot Image		
Paste Bitmap Size		Insere bitmap do Clipboard na janela
Bitmap Original	Size	Coloca bitmap no tamanho original.
Edit Bitmap		
Select All	F2	Seleciona todos os objetos da janela.
Links		Mostra menu de Links
Cut Links		Copia todas as animações do objeto selecionado para o Clipboard e apaga-as.
Copy Links		Copia todas as animações do objeto selecionado para o Clipboard.
Paste Links		Copia as animações do Clipboard para o(s) objeto selecionado(s).
Clear Links		Elimina todas as animações do(s) objeto(s) selecionado(s).
Enlarge Radius	Shift+Mais	Aumenta o raio do (s)objeto(s) selecionado(s). (retângulo com bordas arredondadas)
Reduce Radius	Shift+Menos	Reduz o raio do(s) objeto selecionado(s) (retângulo com bordas arredondadas).
Reshape Object	Ctrl+R	Permite redesenhar o objeto selecionado. (polígonos e linhas múltiplas).
Add Point		Adiciona pontos a polígonos e linhas múltiplas.
Del Point		Elimina pontos de polígonos e linhas múltiplas.

Arrange

COMANDO	TECLAS DE ATALHO	AÇÃO
Send to Back	F9	Posiciona o(s) objeto(s) selecionado(s) atrás de todos os outros objetos da janela.
Bring to Front	Shift+F9	Posiciona o(s) objetos selecionados na Frente de todos os outros objetos da janela.
Align		Mostra menu de alinhamento.
Align Left	Ctrl+F3	Alinha os objetos selecionados à esquerda.
Align Center	Ctrl+F5	Alinha os objetos selecionados ao centro (vertical)
Align Right	Ctrl+F7	Alinha os objetos selecionados à direita.
Align Top	Ctrl+F4	Alinha os objetos selecionados no topo.
Align Middle	Ctrl+F6	Alinha os objetos selecionados ao centro (horizontal).
Align Bottom	Ctrl+F8	Alinha os objetos selecionados na base.
Align Center Points	Ctrl+F9	Alinha o ponto central dos objetos selecionados.
Space Horizontal	Crtl+H	Espaça horizontalmente os objetos selecionados.
Space Vertical		Espaça verticalmente os objetos selecionados.
Rotate Clockwise	F6	Gira o(s) objeto(s) selecionado(s) 90 graus no sentido horário.
Rotate CouterClockwise	Shift+F6	Espelha o(s) objeto(s) selecionado(s) na horizontal.
Flip Horizontal	F7	
Flip Vertical	Shift+F7	Espelha o(s) objeto(s) selecionados na vertical.
Make Symbol	F5	Agrupa os objetos selecionados como símbolo.
Break Symbol	Shift+F5	Desagrupa o símbolo.
Make Cell	F3	Agrupa os objetos selecionados.
Break Cell	Shift+F3	Desagrupa os objetos selecionados.
Snap to Grid		Coloca/Retira o Grid.

Text

COMANDO	TECLAS DE ATALHO	AÇÃO
Fonts...		Mostra dialog-box para seleção de fonte.
Bold		Seleciona negrito/não negrito.
Italic		Seleciona modo itálico/não itálico.
Underline		Seleciona modo grifado/não grifado.
Left Justified		Alinha texto à esquerda.
Centered		Centraliza o texto.
Right Justified		Alinha o texto à direita.
Enlarge Font	Ctrl+Mais	Aumenta o tamanho da fonte.
Reduce Font	Ctrl+Menos	Reduz tamanho da fonte.

Special

COMANDO	TECLAS DE ATALHO	AÇÃO
Security		Seleciona comandos referentes à segurança.
Animation Links	Ctrl+A	Mostra dialog-box para animação do objeto selecionado.
Substitute Tags...	Ctrl+E	Troca nome(s) do(s) tag(s) do(s) objeto(s) selecionado(s).
Substitute Strings...	Ctrl+L	Troca conteúdo do(s) string(s) selecionado(s).
Tagname Dictionary	Ctrl+T	Acesa o dicionário de dados.
Alarm Groups...		Cria ou modifica grupos de alarme.
Access Names...		Cria ou modifica grupos DDE.
Cross Reference		
Notify Clients		
Configure...		Mostra dialog-box para configuração.
Update Use Counts ...		Atualiza contadores de tags.
Delete Unused Tags ...		Mostra dialog-box para eliminar tags não utilizados Scripts Mostra menu dos possíveis scripts.
Scripts		
Aplication Scripts...		Permite editar script da aplicação.
Window Scripts		Permite editar script da janela
Key Scripts...		Permite editar script associado a uma tecla.
Condition Scripts		Permite editar script associado a tag ou expressão
Data Change Scripts ...		Permite editar script associado à mudança de valores de tags.
TemplateMaker		
SPC		
SQL Access Manager		

Help

COMANDO	Ação
Index	Mostra o índice do Help.
Using Help	Explica como o Help deve ser usado.
About Supervisório	Mostra informações a respeito do Supervisório
Windows	Windows, como número da versão e direitos de cópia.

Windows: Mostra as janelas abertas no momento.

- **Window Properties Ctrl+W**:Mostra dialog-box para alteração das propriedades da janela.

Show Tag Count [nº tags usadas]: Mostra a quantidade de tagnames definidos atualmente pela aplicação (não inclui as variáveis do sistema)

View

COMANDO	TECLAS DE ATALHO	AÇÃO
General Toobar		Vizualiza a Barra de Ferramenta Geral
Wizards/AxtiveX Toobar		Vizualiza a Barra de Ferramenta Wizard
Format Toobar		Vizualiza a Barra de Ferramenta formatação
Draw Object Toobar		Vizualiza a Barra de Ferramenta Desenho
View Toobar		Vizualiza a Barra de Ferramenta Visual
Arrange Toobar		Vizualiza a Barra de Ferramenta arranjos
Status Toobar		Vizualiza a Barra de Ferramenta Status
Ruler		Vizualiza a Barra de Ferramenta Régua
Application Explorer		Vizualiza a Barra de Ferramenta Explorer
Hide All		Esconde todas as barras de ferramentas com exceção da barra de menu e barra de Título.
Full Screen		Esconde todas as barras de ferramentas inclusive a barra de menu E Titulo.

Runtime! Transfere rapidamente do Window Maker parao Window Viewer. Todas as alterações feitas nas janelas abertas no Window Maker serão salvas antes da transferência para o Window Viewer (não pede confirmação!).

Development! Transfere rapidamente do Window Viewer o Window Maker.

5.3.5- DESENVOLVENDO APLICAÇÕES

Variáveis do Intouch

Cada objeto no *Intouch* permite um conjunto de animações, basta selecioná-las ou não. A animação precisa ser configurada. Por exemplo, se desejarmos um indicador em barra do nível de um determinado tanque, primeiro temos que desenhar o indicador. Sobre o local onde desejamos a indicação, clicamos duas vezes (*double click*). Um conjunto de animações estará disponível.

O objetivo é um enchimento percentual do retângulo do indicador, logo a animação é Percent Fill Vertical. Mas o supervisório necessita da informação de qual variável será indicada e esta variável receber informações do transmissor de nível do referido tanque.

Uma variável nada mais é do que um espaço de memória que foi alocado. Esta variável basicamente pode ser dos tipos indicados abaixo.

TIPO	DESCRIÇÃO	EXEMPLO DE APLICAÇÃO
Discrete	Variável que possui apenas dois níveis, 0 ou 1, ativada ou não ativada, ligado ou não ligada.	Bombas, válvulas on/off, lâmpadas, alarmes, etc.
Integer	Variáveis inteiras, ou seja, números inteiros (conjunto Z)	Indicações inteiras, saídas inteiras, etc.
Real	Variável real, ou seja, conjunto R	Indicações reais, saídas reais, etc.
Message	Variável alfanumérica, acumula números e/ou letras.	Informações que podem ser números e/ou letras.
Group Var	Grupo de variáveis, as variáveis podem ser agrupadas para melhorar a organização ou até mesmo para alarmar em uma janela de alarmes.	Alarmes, organização, etc.
Hist Trend	Variável do gráfico de tendência histórica. Cada gráfico necessita de uma. O gráfico de tendência histórica é o equivalente ao registrador, com as vantagens da carta não "embolar", a tinta não "borrar" ou acabar, não ocupar espaço no painel, além de poder ser criado diversas vezes.	Gráfico de tendência histórica e wizard.

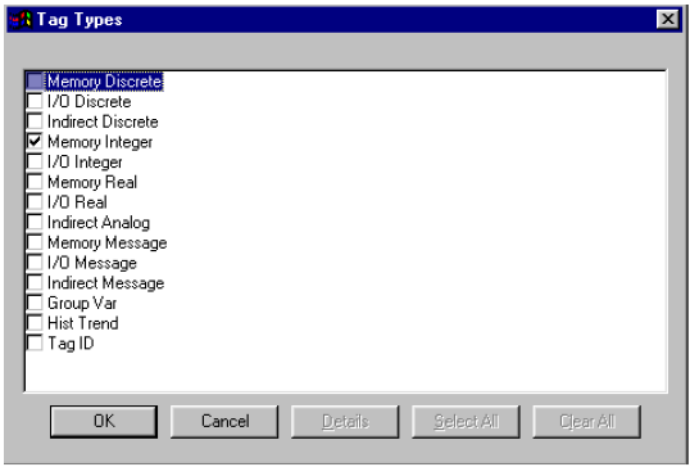

Figura 5.9 - Caixa de diálogo da versão 7.0 e 7.1 do Intouch

No *Dialog box* acima temos os diferentes tipos de variáveis. Ele aparecerá toda a vez que for criada uma nova variável.

O espaço de memória destinado ao acúmulo de informações será chamado de variável e que no *Intouch* será chamado de *tagname*, que pode ser **memory** ou **I/O**. Para cada CLP de fabricante e modelo diferente, existe um *driver* desenvolvido com os comandos para acessar o mapa de memória do equipamento.

O software de *Window Viewer* efetua comunicação com o driver e este, através de uma interface de comunicação, comunica-se com o CLP, o controlador, o barramento. etc.

O *Intouch* diferencia tagname maiúsculo de minúsculo, ou seja, o tagname bomba é diferente do tagname BOMBA.

Animação

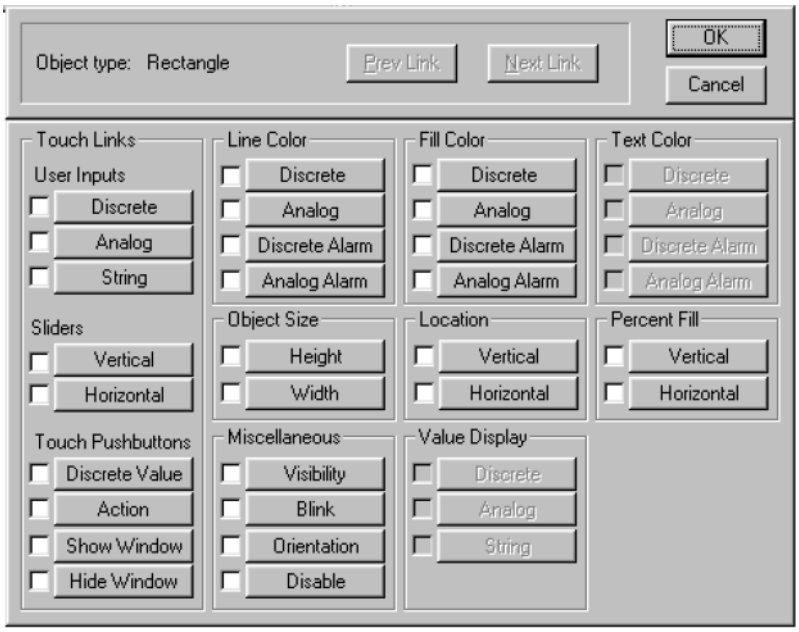

Figura 5.10 - Tela de definições de animação do Intouch

Object type: informa qual o objeto será animado.

Touch Links: Animação ao clicar ou arrastar.

- **User Inputs:** Caixa de entrada de valor.
- **Sliders:** Barra de movimentação.
- **Touch Pushbuttons:** Animação em botão.
    - **Discrete Value :** Usado para variáveis digital.
        - **Direct**: valor real do bit
        - **Reverse**: valor invertido do bit
        - **Toggle:** inverte o valor que estiver o bit 0/1 1/0.
        - **Reset:** O bit vai para 0.
        - **Set:** O bit vai pra 1.
    - **Action**: Linha de programação (script).
    - **Show Window:** Abre janela(s).
    - **Hide Window:** Fecha janela(s).

- **Line Color:** Cor da linha.
- **Fill Color:** Cor do objeto.
- **Text Color:** Cor do texto.
- **Object size:** Tamanho do objeto. (Sempre variável real ou inteira)
- **Location:** Localização do objeto na tela. (Sempre variável real ou inteira)
- **Percent Fill:** O preenchimento do objeto. (Sempre variável real ou inteira)
- **Miscellaneous:**
    - **Visibility:** Objeto visivel.
    - **Blink:** Pisca o objeto.
    - **Orientation:** Rotaciona o objeto.
    - **Disable:** Desabilita as outras animação que o objeto tiver.
- **Value Display:** Informa o valor que a variável possui.

Diferenças:

- Sliders e Location: o primeiro pode ser arrastado pelo mouse o outro e a partir do valor da variável.
- Value Display e User Inputs: o primeiro indica o valor da variável o outro o usuário entra com o valor.

Scripts

O *Intouch* possui um tipo de linguagem que o torna muito mais versátil, está linguagem é chamada de Script, tem uma certa semelhança com “Pascal”, “C” e “Basic”.

O Script pode ser:

TIPO	DESCRIÇÃO
Application	Será executado enquanto o *Window Viewer* estiver ativo, entrar na memória ou sair da memória. O período será determinado em mseg.
Windows	Script da janela, que será executado enquanto a janela que o possui abrir, fechar ou estiver ativa.
Key	Script de teclado, que será executado quando determinadas teclas forem ou estiverem sendo pressionadas.
Condition	Script que será executado quando ou enquanto determinada condição for verdadeira ou falsa.
Data Change	Script que será executado na mudança de dado de uma determinada variável.

Os comandos da linguagem Script é os mais diversos, vãos desde a atribuição de valores até a exclusão de arquivos.

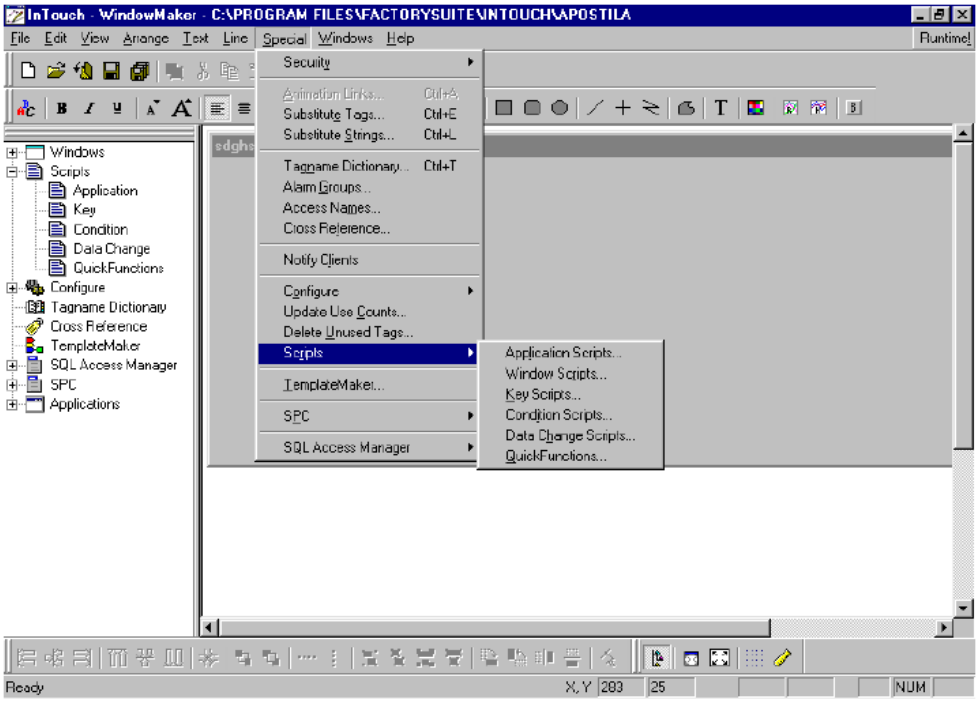

Figura 5.11 - Menu de Scripts do Intouch

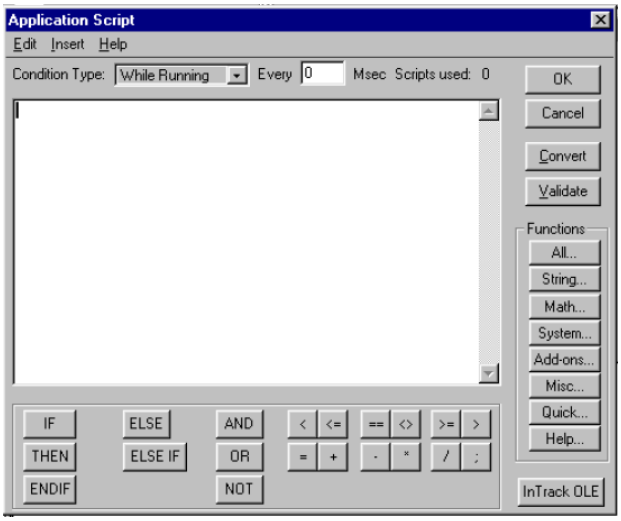

Figura 5.12 - "dialog box" para a edição do script da aplicação

Registros

Basicamente o *Intouch* possui dois tipos de registros, o gráfico de tendência real e o gráfico de tendência histórica.

O gráfico de tendência real mostra dinamicamente as mudanças que estão ocorrendo das variáveis registradas no período de tempo determinado. Este registro funciona na memória Ram do computador, ou seja, não é armazenada para posterior análise. Cada gráfico pode registrar até quatro variáveis simultaneamente.

O gráfico de tendência histórica mostra estaticamente as variáveis registradas. O período, o dia, a hora, o minuto e o segundo são determinados pelos campos de entrada. As informações são armazenadas no disco rígido do microcomputador ou em rede. São dois os arquivos de registro; exemplo: 97062700.log e 97062700.idx, o primeiro é o arquivo que armazena os dados do registro, sendo aammdd00.log (ano, mês, dia) e o segundo é o arquivo de índice do primeiro. O arquivo é aberto com a data do dia do registro.

Alarmes

O *Intouch* possui dois tipos de janelas de alarmes, o sumário de alarmes e o histórico de alarmes.

O sumário de alarmes alarma as ocorrências do momento, já o histórico de alarmes (conhecido como registrador de eventos) registra todas as ocorrências de alarme no disco rígido. A janela de alarme pode alarmar por grupo de variáveis ou por prioridade.

Para separar as ocorrências de alarmes, podemos definir prioridades diferentes para as variáveis e definir no histórico ou sumário de alarmes apenas as prioridades de interesse. Podemos também definir uma variável para o campo de prioridade do histórico ou sumário de alarmes e um botão que mudará a prioridade da janela de alarmes.

A separação das variáveis no histórico ou sumário de alarmes, também pode ser feira pelos grupos de variáveis criados.

Backup do aplicativo

O backup dos arquivos importantes dos discos rígidos é uma prática de bom senso.

Para fazer uma cópia de segurança do aplicativo, copie todo o diretório, não copie os arquivos *.?BK, *.AEH, *.LOG, *.IDX e *.ALG.

Para fazer backup dos registros basta copiar apenas os arquivos *.log e *.idx. A cópia pode ser feita pelo gerenciador de arquivos do *Windows*. A cópia pode ser compactada com winzip, ou qualquer outro software que faça compactação de arquivos.

Os arquivos *.AEH são relatórios, em formato ASCII, das ocorrências do *Intouch*.

5.4- EXEMPLO DE UTILIZAÇÃO DO INTOUCH

As atividades a seguir visam exercitar as ferramentas do *Intouch*. O objetivo final é o desenvolvimento de duas telas com navegação entre elas. O objetivo deste capítulo é criar uma nova aplicação com o nome telas uma janela com o nome processo.

5.4.1- CRIANDO UMA NOVA APLICAÇÃO

Problema 1 - Criar uma nova aplicação com o nome telas e uma janela com o nome processo.

1. Crie uma nova aplicação (novo diretório)
   1.1. Carregue o gerenciador de aplicativos do InTouch
2. Create Directory: "telas"

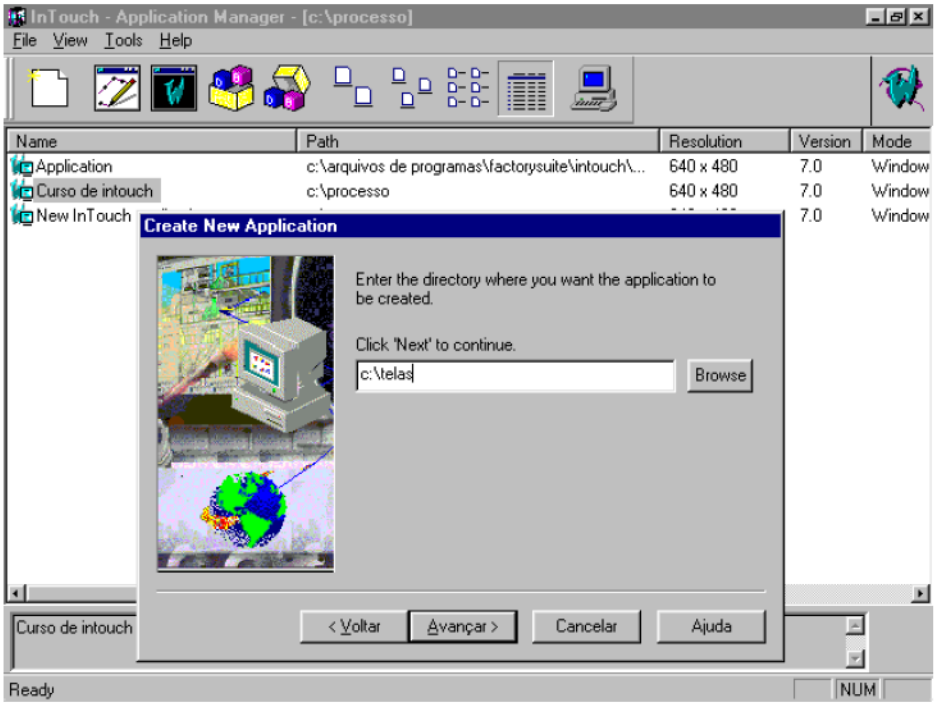

Figura 5.13 - Tela de criação de nova aplicação Intouch

3. Crie uma nova janela
   3.1. File/New Window ou botão NEW do WindowMaker
   3.2. Window Name:"processo"

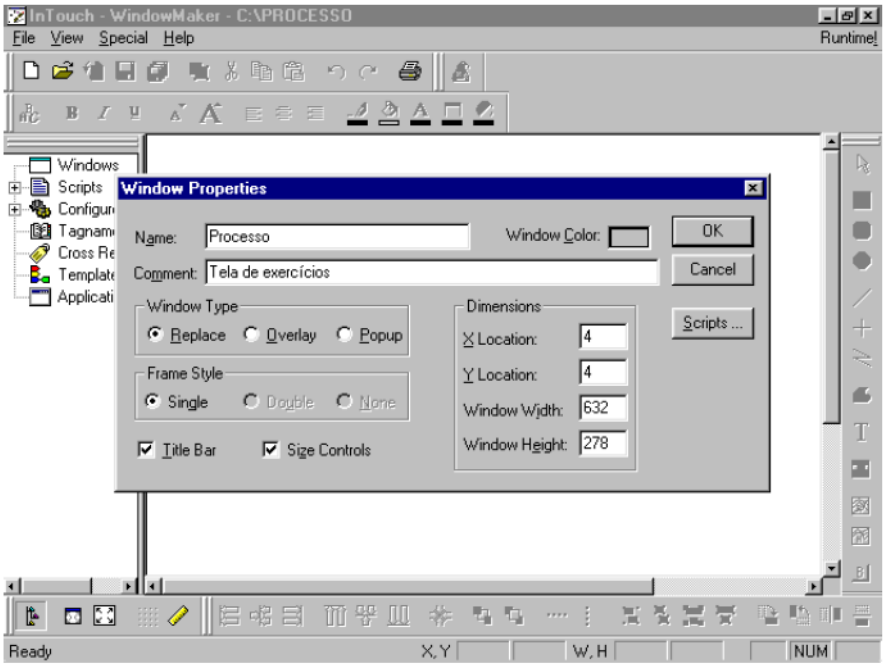

Figura 5.14 - Propriedades de janela do problema 1

5.4.2- GRÁFICO DE BARRAS UTILIZANDO VARIÁVEL DO SISTEMA

Problema 2 - Desenhar e animar um indicador que indique os segundos do microcomputador.

1. Desenhe um retângulo.
2. Desenhe uma régua com escala de 0 a 60.

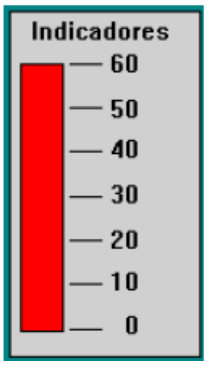

Figura 5.15 - Régua com escala de 0 a 60.

3. *Double-click* no retângulo de indicação.
4. Selecione a animação *PERCENT FILL VERTICAL*.

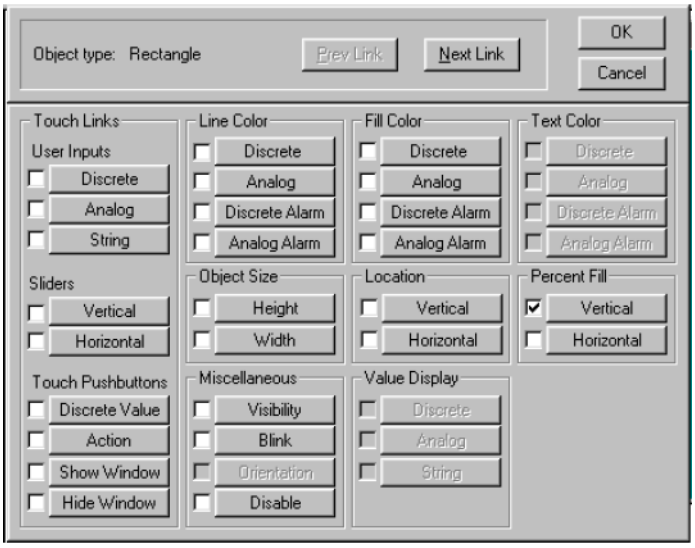

Figura 5.16 - Propriedades de animação do problema 2

5. No campo *Expression* dê um *double-cl ick* para selecionar uma tag do banco de dados.
6. Selecione a variável do sistema "$Second"
7. Complete a tela de animação com
   - Value at Max Fill: 60
   - Value at Min Fill: 0
   - Max % Fill: 100
   - Min % Fill: 0
   - Direction: Up

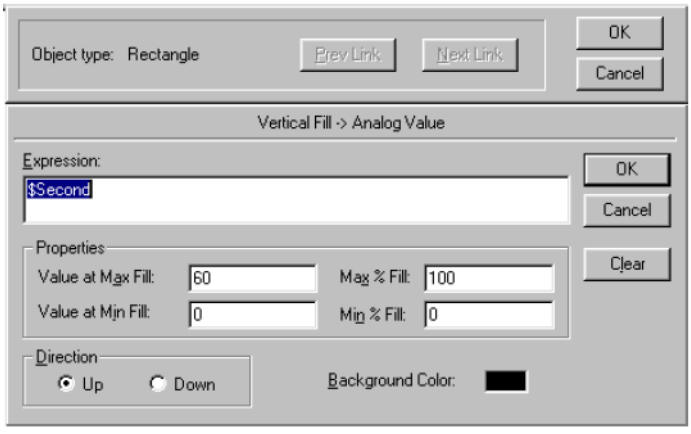

Figura 5.17 - Configurando variáveis do problema 2

8. Selecione o comando Runtime, para verificar a animação do gráfico de barras.

5.4.3- BOMBA – BOTÃO LIGA/DESLIGA – SAÍDA COM ESTADO DA BOMBA

Problema 3 - Criar uma bomba que mudará de cor em função do seu estado (verde para ligada e vermelha para desligada), um botão para ligá-la e desligá-la e um campo que mostrará o texto "bomba desligada" quando a bomba estiver desligada e "bomba ligada" quando a bomba estiver ligada.

1. Desenhe uma bomba

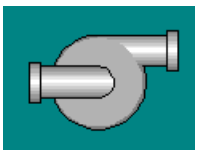

Figura 5.18 – Desenho da bomba do problema 3

2. Agrupe as figuras em um símbolo.
3. Double-click no símbolo
4. Selecione a animação FILL COLOR DISCRETE.

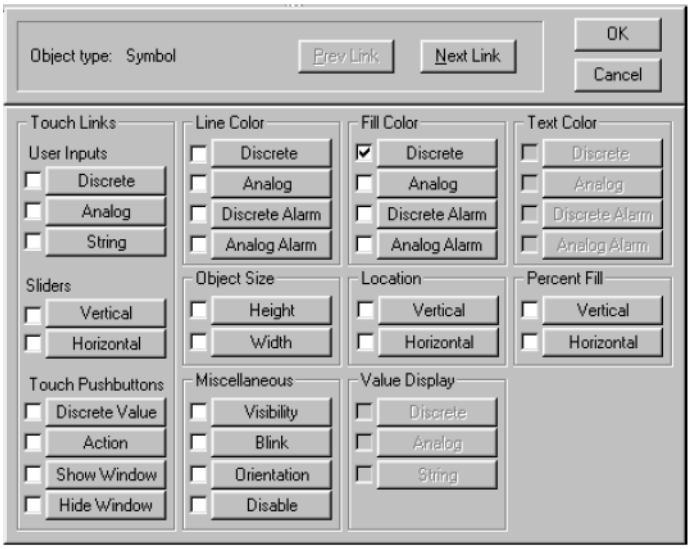

Figura 5.19 - Propriedades de animação da bomba

5. Crie o tag bomba, tipo Memory Discrete.

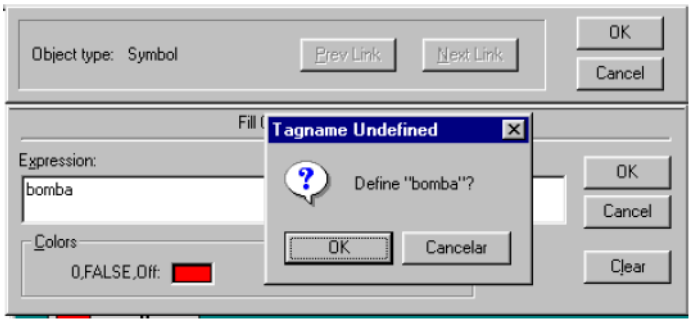

Figura 5.20 - Definição da tagname bomba

- Esta é um tagname que não existe, logo o Intouch questiona se deseja criar.

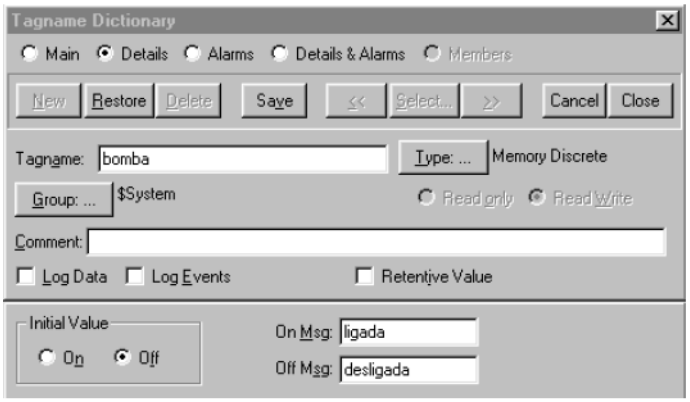

Figura 5.21 - Configuração da tagname bomba

6. Complete a tela de animação com:
   - OFF _ vermelho (cor da bomba no estado desligado)
   - ON_ verde (cor da bomba no estado ligado).

Figura 5.22 - Configurando variável bomba

7. Crie botão "Liga/Desliga"

Figura 5.23 - Criação do botão liga/desliga do problema 3

8. Double click no botão.
9. Selecione animação TOUCH PUSHBUTONS - Discrete Value.

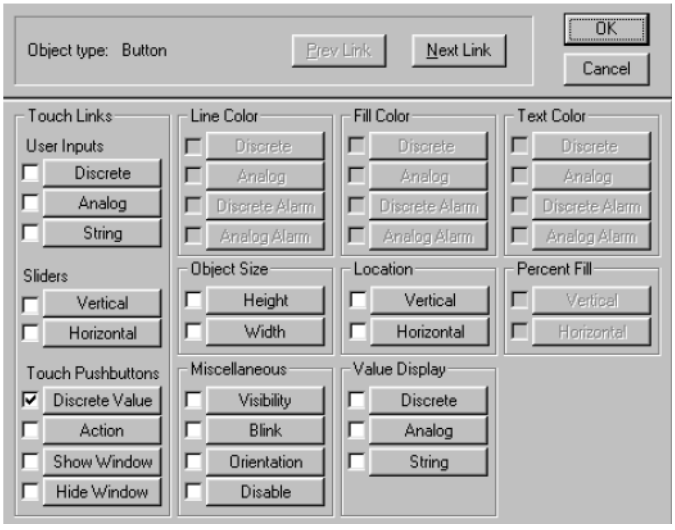

Figura 5.24 - Propriedades de animação do botão liga/desliga

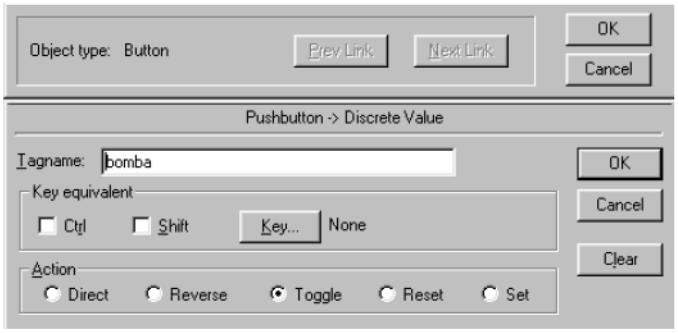

Figura 5.25 - Configuração da variável botão liga/desliga

10. Complete a tela de animação com
    - tag: BOMBA (tag já criada anteriormente)
    - action: TOGGLE
11. Crie um texto "#"

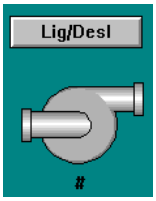

Figura 5.26 - Criação de texto no problema 3

12. Double-click no texto "#"
13. Selecione animação VALUE DISPLAY – DISCRETE

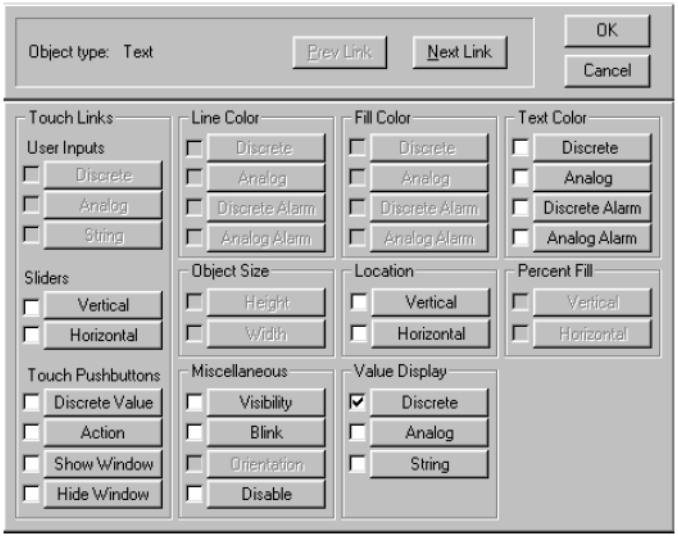

Figura 5.27 - Propriedades de animação do texto

14. Complete a tela de animação com
    - tag: BOMBA
    - ON Message: "bomba ligada" (mensagem a ser mostrada quando a bomba estiver ON)
    - OFF Message: "bomba desligada" (mensagem a ser mostrada quando bomba a estiver OFF)

Figura 5.28 - Configuração da variável texto

5.4.4- GRÁFICO DE BARRAS – VALOR ENTRADO PELO OPERADOR – BOTÃO

Problema 4 - Criar um indicador de set-point e um botão ao lado que faça o ajuste através de DRAG e um que faça o ajuste do set-point através do teclado.

1. Double-click no Retângulo do gráfico de barras.
2. Click na opção de animação PERCENT FILL VERTICAL para alterar a animação feita anteriormente com variável do sistema.
3. No campo Expression digite SETPOINT, novo tag a ser criada.
4. Crie o tag SETPOINT do tipo Memory Integer, com os seguintes parâmetros:
   - Min Val: 0
   - Max Val: 60
5. Crie um texto "Setpoint = #", que vai ser um campo de entrada.
6. Double-cl ick no texto acima.
7. Selecione a animação VALUE INPUT ANALOG.
   - tag: SETPOINT
   - Min Value: 0
   - Max Value: 60
8. Fazer um triângulo ao lado da régua. Esta figura será um botão de deslocamento.
9. Double-cl ick no triângulo acima.
10. Selecione a animação VALUE SLIDER VERTICAL
    - Tag: SETPOINT
    - Value
    - At Top: 60 At. Bottom: 0
    - Vertical Movement
    - Up: 100 Down: 0
    - Obs: os valore up e down correspondem ao deslocamento do objeto em pixel.

11. Selecione a linha vertical da régua e verifique seu tamanho.
12. Selecione novamente a animação do triângulo e preencha o campo vertical
    - Movement Up com o tamanho da linha.

5.4.5- TEXTO – VISIBILIDADE E BLINK

Problema 5 - Criar um texto "Fogo na área 1" e um botão fogo. O texto deverá aparecer piscando enquanto o botão estiver sendo pressionado, caso contrário o texto desaparecerá.

1. Crie um texto "Fogo na área 1".
2. Selecione a animação do texto VISIBILITY.
3. No campo Expression digite FOGO, novo tag a ser criado.
4. Crie tag FOGO tipo Memory Discrete.
5. Complete a tela de animação com:
    - Visibility State: ON
6. Crie botão com a legenda "Fogo".
7. Selecione a animação do botão TOUCH PUSHBUTTONS DISCRETE.
8. Complete a tela de animação com:
    - tag: Fogo
    - action: Direct
9. Double-cl ick no texto "fogo na área 1".
10. Selecione animação do texto BLINK.
11. Complete a tela de animação com.
    - tag: Fogo
    - blink visible with Text Color: Vermelho

5.4.6- DETECTOR DE GÁS – TAMANHO DO OBJETO

Problema 6 - Criar uma elipse que mudará de largura em função da concentração de gás na atmosfera, a variável da concentração será o próprio SETPOINT.

1. Desenhe uma linha horizontal e sobre ele uma elipse.
2. Selecione a animação da elipse OBJECT SIZE WIDTH.
3. Complete a tela de animação com:
    - Expresion: SETPOINT

- Value at Max Width: 60
- Value at Min Width: 0
- Max % Width : 100
- Min % Width: 0
- Anchor: center

4. Entre no Window Viewer, altere o setpoint e verifique o efeito no tamanho da elipse.

5.4.7- ESTEIRA

Problema 7 - Criar um quadrado sobre uma linha horizontal, o mesmo deverá se deslocar sobre a linha em função dos segundos do micro computador.

1. Desenhe uma linha horizontal.
2. Sobre o canto esquerdo da linha, desenhe um quadrado.
3. Selecione a animação do quadrado LOCATION HORIZONTAL.
4. Complete a tela de animação com
   - Expression: $second
   - Value
   - At Left End: 0
   - At Right End: 60
   - Horizontal Movement
   - To Left: 0
   - To Right: 100
   - Obs: o valor to left e right corresponde ao deslocamento em pixel.
5. Verifique o tamanho do quadrado e da esteira.
6. Acabe de completar a tela de animação do quadrado.
   - Horizontal movement
   - to Right: (tamanho da esteira) - (tamanho do quadrado)

5.4.8- AJUSTE FINO PARA O GRÁFICO DE BARRAS

Problema 8 - Criar um botão para incremento do set-point em uma unidade e um outro para decremento do set-point em uma unidade. Não utilize o botão já pronto ou o wizard.

1. Desenhe o botão de ajuste fino.
2. Selecione a animação deste botão TOUCH PUSHBUTTONS ACTION.
   - Faça o Script. Selecione o tipo (On Key Down)
   - setpoint = setpoint + 1;
   - if setpoint > = 60
   - then setpoint = 60;
   - endif;
3. Entre no Window Viewer e click no botão para aumentar o SETPOINT de uma unidade
4. Siga os mesmos passos para fazer o ajuste fino para diminuir SETPOINT.

5.4.9- UTILIZANDO WIZARD – BOTÃO LIGA/DESLIGA

Problema 9 - Utilizar o wizard para criar um botão de liga/desliga a bomba. Utilize o botão de forma que quando a bomba estiver ligada o mesmo fique verde com o texto interno "desliga" e quando desligada o texto interno deverá ser "liga".

1. Selecione o ícone WIZARD no ToolBox
2. Selecione But tons e o botão START
3. Posicione o botão na janela
4. Complete a animação do botão com
   - Tagname: bomba
   - On Fill Color: verde
   - On Label: desliga
   - Off Label: liga
   - Mude a fonte do texto do botão.

5.4.10- UTILIZANDO WIZARD – BOTÃO INCREMENTA/DECREMENTA

Problema 10 - Utilizar um wizard para incrementar e decrementar o set-point.

1. Selecione o ícone WIZARD no ToolBox.
2. Selecione But tons e o botão Incrementa/Decrementa.
3. Posicione o botão na janela.
4. Complete a animação do botão com:
   - Tagname: setpoint
5. Mude a cor das setas do botão.
6. Mude o parâmetro Increment/Decrementa Value: 2

5.4.11- SCRIPT PARA ANIMAR A ESTEIRA

Problema 11 - Criar um botão "liga/desliga" a esteira que deverá ligar a esteira.

1. Selecione o quadrado sobre a esteira.
2. Modifique a animação LOCATION HORIZONTAL. Substitua a Expression:
   - $second por ESTEIRA
3. Crie o tag ESTEIRA tipo Memory Integer.
   - Min Value: 0
   - Max Value: 60
4. Crie um botão com a legenda "Liga/Desliga".
5. Selecione a animação do botão TOUCH PUSHBUTTONS DISCRETE.
   - tag: LIGA
   - action: Toggle
6. Crie tag LIGA tipo Memory Discrete.
7. Selecione o comando Special/Application Scripts.
8. Faça o script. Selecione a opção While Running every 1000 Msec if liga

```
if liga == 1
        then esteira = esteira + 1;
        if esteira > = 60
                then esteira = 0;
        endif;
endif;
```

9. Entre no Window Viewer e verifique a movimentação da esteira.

5.4.12- OBJETO FEITO COM OUTRO UTILITÁRIO E TRAZIDO COMO BITMAP

Problema 12 - Fazer um desenho no PaintBruch e cole na janela processo.

1. Minimize a aplicação Window Maker.
2. Selecione Acessórios.
3. Selecione Paintbrush.
4. Desenhe qualquer objeto.
5. Copie o objeto para o Cl ipboard.
6. Selecione o ícone BITMAP no ToolBox.
7. Faça um retângulo para determinar a área do Bitmap.
8. Selecione Edit/Paste Bitmap.
9. Selecione Edit/Bitmap original size, para o objeto assumir seu tamanho original.

5.4.13- JANELA DE MENSAGEM PARA O OPERADOR

Problema 13 - Criar uma janela "mensagem" para avisar ao operador que a pressão de um tanque está muito alta. A mesma deverá aparecer sobre a janela de processo. Deverá existir um botão na janela mensagem para ocultar a janela de mensagem e na janela de processo deverá existir um botão para chamar a janela mensagem**.**

1. Crie uma janela Mensagem (File/New Window).
   - tipo da janela : POPUP
2. Escreve na janela mensagem: "Pressão do tanque muito alta".
3. Crie na janela mensagem um botão com legenda "OK".
4. Selecione a animação do botão HIDE WINDOW.
   - Selecione a janela a ser escondida ao clicar o botão: Mensagem
5. Redimensione a janela mensagem e posicione sobre a janela processo.
6. Crie na janela processo um botão com legenda "Mensagem".
7. Selecione a animação do botão SHOW WINDOW.
   - Selecione a janela a ser mostrada ao clicar o botão: Mensagem.

5.4.14- GRÁFICO DE TENDÊNCIA REAL

Problema 14 - Criar uma nova janela “Registro” e nesta crie um gráfico de tendência real com o registro das seguintes variáveis: esteira e setpoint.

1. Crie uma nova janela "Tendência"(File/New Window).
   - tipo da janela: REPLACE
2. Selecione o ícone da tendência real do Toolbox e defina uma área para a mesma.
3. Double-click na figura para entrar na tela de configuração
   - Time Span: 3 min
   - Sample: 2 sec
   - Pen: setpoint
4. Entre no WindowViewer e veja o gráfico de tendência real sendo plotado.

5.4.15- GRÁFICO DE TENDÊNCIA HISTÓRICA

Problema 15 - Criar um gráfico de tendência histórica com as variáveis: esteira e set-point.

1. Selecione o ícone correspondente à Tendência História no Toolbox e defina uma área para a mesma.
2. Double-click na figura para entrar na tela de configuração.
3. Para configurar a tendência histórica não pode estar com o WindowViewer ativado.
4. Selecione CTRL-ESC para finalizar o WindowViewer.
5. Double-cl ick na figura para entrar na tela de configuração.
   - Tag: histórico
   - Time Span: 10 min
   - Min Value: 0 Max: 100
   - Pen: esteira
   - Setpoint
6. Crie o tag histórico do tipo Hist Trend.
7. Modifique os tags esteira e setpoint, selecionando a opção Log Data.
8. Selecione o comando Special/Configure/Historical Logging.
9. Selecione a opção Logging Enabled.
10. Entre no WindowViewer, abrindo a janela Processo.
11. Ligue a esteira e altere valores de setpoint.

12. Mude para a janela Registro
13. Click no gráfico de Tendência Histórica:
    - coloque a hora correta
    - chart length: 10 min

5.4.16- JANELA DE ALARMES

Problema 16 - Criar uma janela de alarmes com um sumário de alarmes, em que deverá alarmar bomba desligada, setpoint muito baixo (abaixo de 5), baixo (abaixo de 10), alto (abaixo de 50) e muito alto (abaixo de 55) e esteira com desvio de 10% para em menos e 20% para mais com origem em 30.

1. Crie uma nova janela "Alarmes" (File/New Window)
    - tipo da janela: REPLACE
2. Selecione o ícone correspondente a Alarme no Toolbox e defina uma área para a mesma.
3. Defina as faixas de alarmes para os tags bomba, setpoint e esteira.
    - BOMBA: alarm state off
    - SETPOINT: Lolo: 5, prioridade: 1
    - Low: 10, prioridade: 5
    - High: 50, prioridade: 5
    - HiHi: 55, prioridade: 1
    - ESTEIRA: MinorDev 10% prioridade: 5
    - MajorDev 20% prioridade: 1
    - Target: 30
4. Double-cl ick na figura para entrar na tela de configuração de alarmes.
    - Inicialmente usando os valores default.
5. Entre no WindouwViewer, abrindo a janela processo.
6. Ligue a esteira, altere o estado da bomba.
7. Entre na janela ALARMES e observe os alarmes gerados.

5.4.17- BOTÕES PARA NAVEGAÇÃO ENTRE JANELAS

Problema 17 - Criar botões nas janelas “Processo”, “Alarme” e “Registro” para a navegação entre elas.

1. Crie três botões com as legendas "PROCESSO”, “ALARME" e "REGISTRO”.
2. Anime estes botões com Show Window.
3. Copie para o Cl ipboard e reproduza estes botões nas outras janelas.

5.4.18- CRIAÇÃO DE GRUPOS DE ALARMES - PRIORIDADES

Problema 18 - Criar o grupo de variáveis analog e digi tal , separando a janela de alarmes para as variáveis pertencentes ao grupo analog. Em seguida faça uma janela de alarmes com separação de variáveis por prioridade.

1. Entre no WindowMaker com a tela de alarme.
2. Entre no dicionário e selecione a esteira.
   - Mude o grupo para ANALOG.
3. Crie o grupo ANALOG.
4. Altere o grupo do setpoint para ANALOG.
5. Selecione a bomba e altere o grupo para DIGITAL.
   - Crie este novo grupo.
6. Altere a configuração da tela de alarmes.
   - Alarm Group: ANALOG.
7. Entre no WindowViewer.
8. Observe que somente aparecem os alarmes analógicos.
9. Altere a configuração da tela de alarmes.
   - From Priority: 3
   - To Priority: 10
10. Entre no WindowViewer.
11. Observe que os alarmes de prioridade 1 não aparecem na janela de alarmes.

5.4.19- BOTÃO DE RECONHECIMENTO DE ALARMES

Problema 19 - Criar um botão de reconhecimento de alarmes das variáveis do grupo analog.

1. Crie um botão com legenda "RECONHECE"
2. Selecione a animação deste botão TOUCH PUSHBUTTONS ACTION
   - On Key Down
   - Ack analog;

5.4.20- HISTÓRICO DE ALARMES – BOTÕES PAGUP E PAGDOWN

Problema 20 - Criar um histórico de alarmes com botões para navegar pelas ocorrências de alarmes, um para avançar e outro para recuar.

1. Modifique a configuração da tela de alarmes para Alarm History.
2. Crie botões para percorrer o histórico de alarmes.
3. Double-click na janela de alarmes.
4. Defina as variáveis para página anterior e página posterior.
5. As duas variáveis devem ser criadas com tipo Memory Discrete.
6. Defina os botões para avançar e retroceder páginas.
7. A animação dos botões deve ser Touch Pushbut tons Discrete Value, com ação reset.

REFERÊNCIAS BIBLIOGRÁFICAS

Atos Automação Industrial LTDA; **Curso Básico de Controladores Programáveis,** 2006, Apostila de Curso de Controladores Programáveis.

Silva M. E.; **Automação Industrial,** 2005, Apostila da disciplina de automação industrial das instituições: FUMEP, EEP, COTIP.

Konte J. F., Teixeira J. L. P., Alves P. E.; **Curso de Controladores Lógicos Programáveis,** 2009, Apostila do curso de controladores lógicos programáveis da UERJ.

Antonelli P. L.; **Introdução aos controladores lógicos programáveis (CLPs)**, 1998, Apostila de Curso de CLP Básico.

Corradi., J.; **Controladores Lógicos Programáveis**, 2005, Apostila de Curso de CLP.

HI Tecnologia; **Manual do usuário – Ambiente de Programação SPDSW**, 2010, Manual de instruções do software SPDSW.

Maitelli, A. L.; **Apostila de Controladores Lógicos Programáveis**, Apostila do curso de CLP do Departamento de Engenharia de Computação e Automação da UFRN.